FORSCHUNGSZENTRUM JÜLICH GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich, Tel. (02461) 61-6402

Interner Bericht

Massively Parallel Computing Using the Intel Paragon XP/S Computer at the Central Institute for Applied Mathematics of the Research Centre Jülich (KFA)

Heribert C. Burg (ed.)

KFA-ZAM-IB-9609

Mai 1996
(Stand 29.05.96)

Dieser Bericht wurde erstellt als abschließender Sachbericht zum Fördervorhaben ITH9301 des BMBF.

Preface

This report is a collection of different applications implemented and running on the massively parallel computer Intel Paragon XP/S at the Central Institute for Applied Mathematics (ZAM) of the Research Centre Jülich (KFA). It shows that the massively-parallel architecture approach has been accepted by the user community while the range of applications covers an enormous spectrum. Meanwhile, massively-parallel computers have entered a stage, where they are able to serve as powerful elements in a production environment. The applications running at ZAM prove that this kind of computers can be used also by industry, now.

Contents

1. Introduction

In present literature concerning high-performance computing we often hear of so-called "Grand Challenges" [7]. These are important applications from quite different sciences, which need orders of magnitude more compute power, than the best systems can deliver today. The methods of accelerating computers used in the past, like improving processor speed, can not realize these multiplication factors in compute power needed. The only remedy is parallel processing, i.e. using many processors simultaneously for the same problem to get the solution as much faster as there are processors which can be used.

Several years ago, parallel processing was done with only a few processors communicating via a shared memory. This meant the user did not need to worry about the data distribution, because each processor looked for the data in the one and only common (shared) memory. Therefore, these systems were rather easy to program, for the organisation of the data was simple from the user's point of view. Though of course programming a parallel computer is more difficult than programming a sequential computer. For the shared-memory computers parallel programming can be done automatically up to a certain level of efficiency. This was the Golden Age of the vector-supercomputers with modest parallelism [5], e.g. CRAY Y-MP or CDC ETA-10.

As there are many interesting applications among the Grand Challenges, especially for industry, one would like to ask why don't we build shared memory machines with thousands of processors, if they are so easy to program. Unfortunally, it is today not possible to build a computer of the shared memory type with, say, more than 64 processors; the problems arising when all these processors work on the same memory are too complex and their technical solutions are too expensive.

Thus at the moment we are confronted with the situation that on one hand we need computers with thousands of processors for only parallel processing brings sufficient speed, on the other hand these machines have to be built with distributed memory, i.e. there is no common memory, each processor has a memory of its own. This makes programmer's life difficult: Before doing computation in any of the processors, he has to take care that the data needed for the computation are already in the local memory. Otherwise, the necessary data has to be transported from a remote memory to the local memory. So messages containing the requested data have to be sent from one processor to another. This transportation or communication between the processors can be very costly. In many cases, often after the first implementation of an algorithm, the programmer has to realize that the communication costs more than what is gained by the use of the many processors. So, there is slow-down instead of speed-up as the result of parallelizing the application.

The only solution to this problem seems to be a user interface for the distributed memory machines, which keeps the problems of dealing with the distributed memory away from the user. Only with a remedy like that to catch up with the problems of distributed memory, big numbers of users and applications from outside computer science will be motivated to use the novel architecture approach of massive parallelism. One of the most promising approaches in this field is the concept of "Shared Virtual Memory" (SVM) [3]. This technique enables the user to program the distributed memory machine as if it were a shared memory computer.

In 1991 KFA installed its first massively-parallel computer, an Intel iPSC/860, a system with 32 nodes in hypercube configuration [4]. Though from the point of view of today one would not regard a 32-processor-machine as a massively parallel one, the problems arising when

changing from the well-known and well-established CRAY number-crunchers of vector-supercomputer type were enormous**.** Programming a parallel machine by doing explicit communication in the form of message passing was a new world of computing.

The iPSC computer at KFA was soon followed by an Intel Paragon XP/S-5 system [2] with 72 compute nodes in 1993. Each compute node consists of two Intel i860XP microprocessors; one is used for computing, the other serves for communication. Contrary to the iPSC-system, the nodes of the Paragon are organized in a two-dimensional grid structure. Figure 1 shows the configuration of the Intel Paragon computer at the Research Centre Jülich.

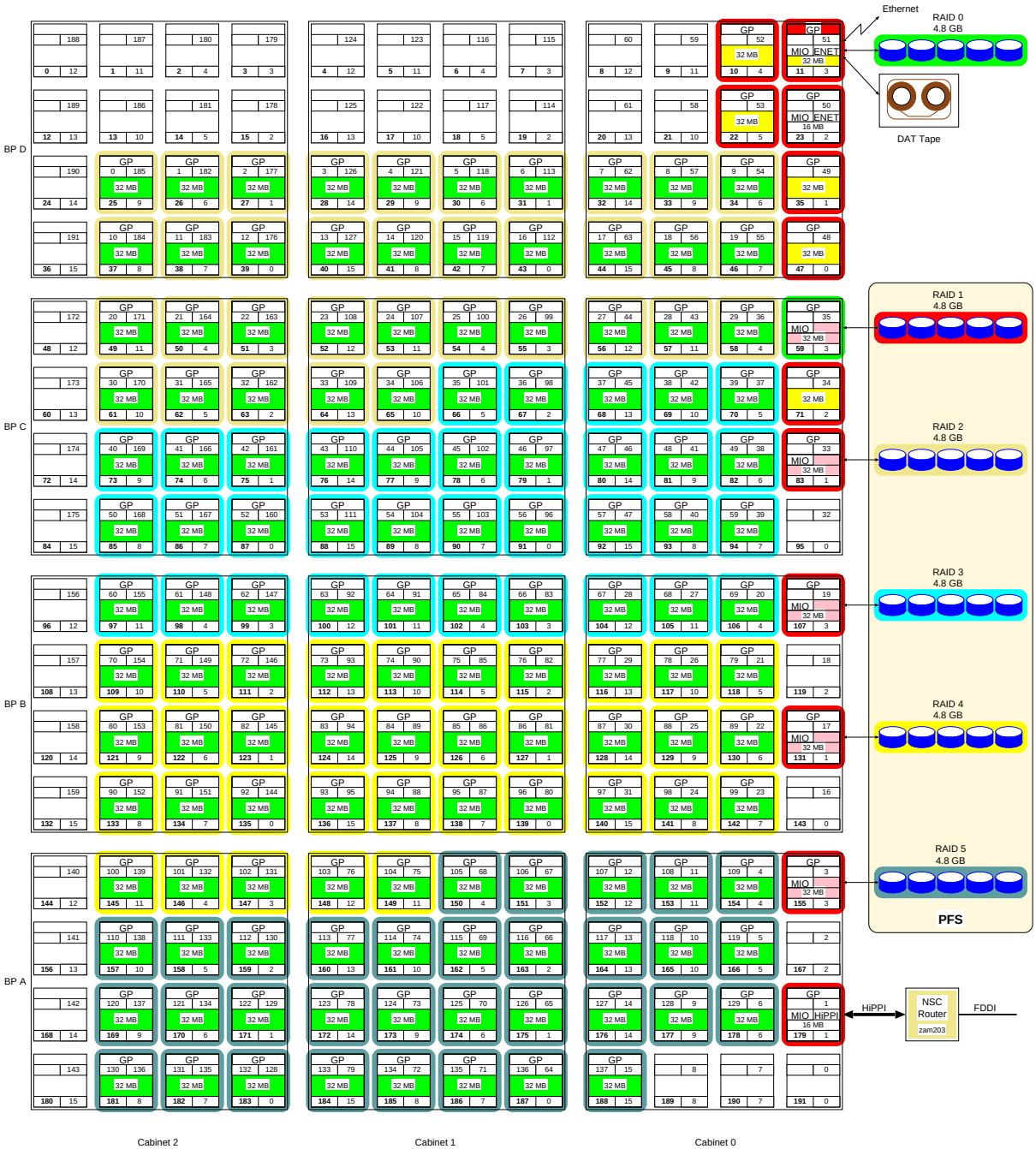


Figure 1: Configuration of the Intel Paragon XP/S-10 at the Research Centre Jülich

In comparison with hypercubes, the grid-like architectures have the advantage that the interconnection complexity grows just linearly, if the number of nodes is increased. The i860-microprocessor has a clock speed of 50 MHz leading to a theoretical peak compute rate of 75 MFLOPS by several interior parallelisms. Now each node has 32 MB of its own private memory. The communication bandwidth of the interconnection network allows theoretical data transfer of 400 MByte per second. Five RAID-Systems (Redundant Array of Independent Disks) with a capacity of 4.8 GByte each serve for mass storage. The external connections include HiPPI and FDDI interfaces.

In February 1994, the German Federal Ministry of Research and Technology (BMFT, now BMBF) supported the KFA Paragon installation, which allowed to double the number of nodes. The system then advanced to an XP/S-10 model with 140 compute nodes. It has a theoretical peak rate of 10 GFLOPS and an accumulated memory of about 4.5 GByte. Later the number of compute nodes was reduced to 138, when it was decided to use some compute nodes as additional service nodes (for compiling and several service modes) to catch up with a bottle neck. It has to be said that the sustained performance of the KFA Paragon, like of all massively parallel systems, is far away from the theoretical numbers [1]. Getting ten percent of the peak node performance is quite normal, so in absence of a theoretical linear speed-up, 700 to 900 MFLOPS is quite a good performance for the machine under consideration. There are, however, applications using assembler code which got up to 3 GFLOPS out of the machine. The sustained communication bandwidth between the nodes is about 60 Mbyte/s.

The financial support of the Federal ministry has to be seen in context with the so-called "Höchstleistungsrechenzentrum" (High Performance Computing Centre, HLRZ). This was founded in 1987 by the three German Research Centres DESY (Deutsches Elektronensynchrotron, Hamburg), GMD (GMD — Forschungszentrum Informationstechnik, Sankt Augustin) and KFA. The HLRZ should present a platform for computational science all over Germany [6]. Universities who could not afford a high performance computer of their own should have the possibility to do supercomputing at a public system, offered by HLRZ. The HLRZ idea has proven very successful and can well serve as a model for the recent recommendation of the German Science Council to establish up to four nation wide very high performance computer centres. Most of the applications listed in the second part of this report are applications which have been (successfully) proposed along the HLRZ regulations. This is also shown in figure 4.

In the first time after the installation, using the Paragon computer was rather discouraging since there were many reboots, sometimes up to ten a day, cf. figure 2. Fortunately, many users were willing to act as pioneers in massively parallel computing. The team at ZAM, especially the members of the operating system group, worked hard to identify the reasons for the reboots, to communicate errors to Intel or to find workarounds for problems not yet fixed. In figure 2 it can be seen that they were very successful indeed during the last two years. Consequently, the number of reboots decreased to the acceptable level we have now. In this context it has to be said that the local Intel parallel systems engineer at KFA was a most busy and helpful actor, while the responsiveness of the maufacturer itself was sometimes disappointing. With the inroduction of new operating system releases the reboot situation each time became critical again. Despite all the difficulties, the number of reboots shrunk from about 30 per week, which is absolutely inacceptable for a production system, to one or two per week, which is the current state.

With the system getting more and more stable, the number of users and different applications grew. The number of users on the machine increased from 20 per month to about 60 per month, cf. figure 2. Correspondingly, system-usage continuously improved. In figure 3 it is

shown that with the exception of the summer holiday months the system-usage rather soon reached the most pleasing value of 70 percent, in December 95 even 80 percent. From the outside point of view of a normal (non-scientific) computing centre with sequential computers this seems to be not quite sufficient. Here, however, in the context of a massively parallel machine, we have to realize that there are 140 compute nodes to be used. Consider the case that there are 80 processors already in use, then a new job applying for 64 processors cannot be started because the number of remaining processors is too small. So 70 percent of usage, that is about 100 processors of the machine in permant use, day and night, working days and holidays, is quite good.

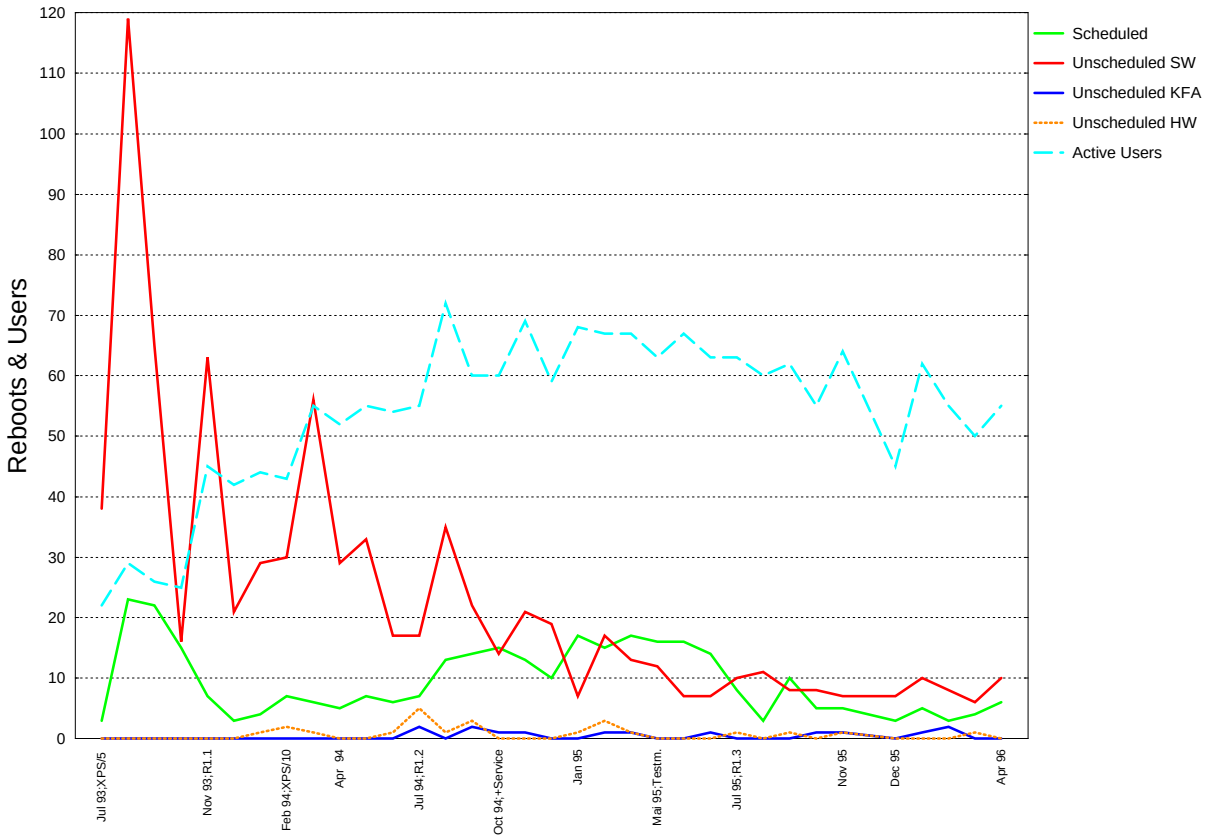


Figure 2: Monthly reboots and active users since installation

Figure 3 makes clear that by far the biggest part of the used time is consumed in batch mode. This is explained by the fact that new applications are tested in interactive mode with only a small number of processors. After the program development has come to an end, the application runs as production code during batch time. Batch time is mainly during the night and at the weekends. There seems to be a rather big portion of time spent on maintenance. Since the installation, a lot of time has been needed to explore the reasons of the reboots. This is done during test periods. These are also used for "critcial" applications, which are suspected to have caused a system breakdown. Thus it has to be said that also a large part of the maintenance time is indeed used for development of applications.

One might expect more continuity in the usage percentage, but it has to be seen that projects have to apply for (new) compute time every six months. So there is some up and down in projects (applications) being in a development or production phase; the former stands for lower, the latter for higher system usage.

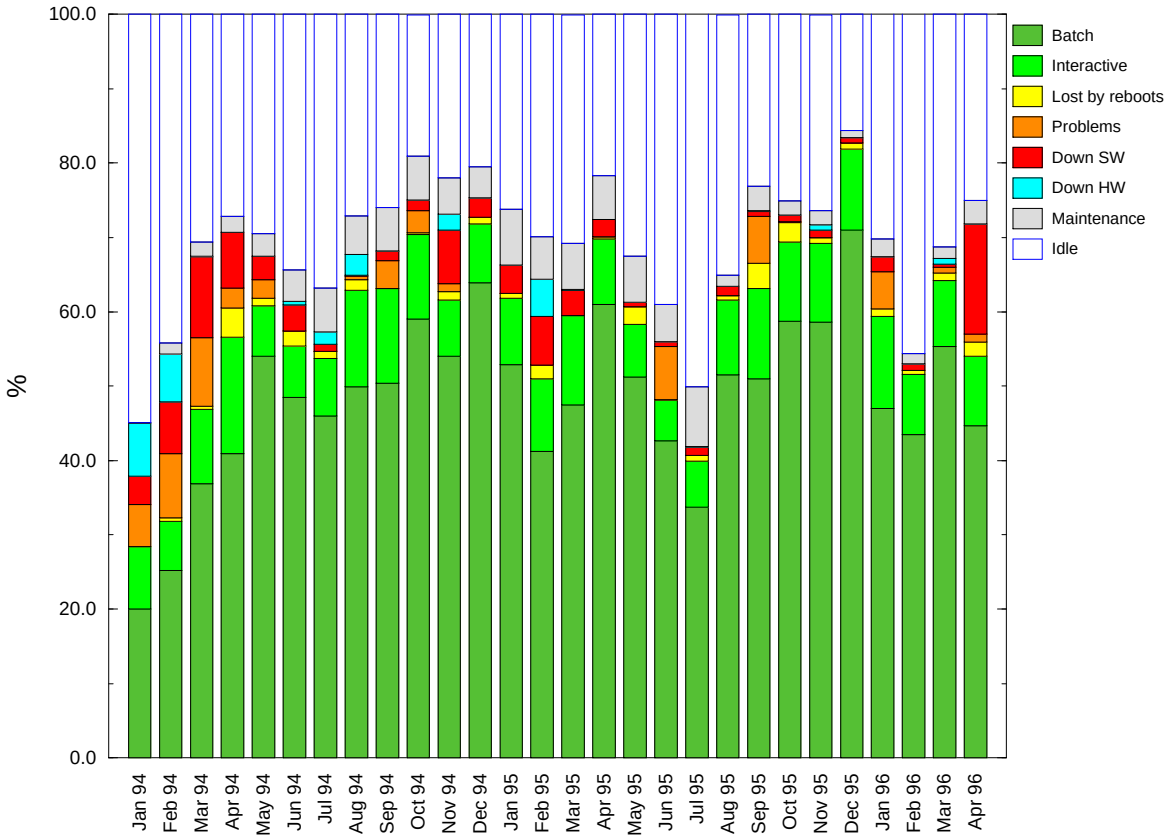


Figure 3: System-usage

Concentrating just on the green parts of figure 3, leads to figure 4. Here, it is shown how the system production time, either batch or interactive, is distributed among the different user groups. Obviously, the users from the HLRZ are by far the best customers of the Intel Paragon at Jülich. Users from the Central Institute for Applied Mathematics itself and users from other institutes of the Research Centre Jülich get smaller portions of the cake. The applications proposed in the context of the HLRZ come from universities and research centres all over Germany, but there are also users from France, the United Kingdom, Austria, Switzerland, Israel and Australia as can be seen in the list of applications, cf. chapter 2. The number of users from industry is yet too small, but along with the EUROPORT activities they will probably grow. There are some industrial companies running codes on the Intel Paragon at Jülich, but this work is mainly done within joint research projects with scientific groups from universities and research institutes. Some research institutes, however, are running code which is of significant interest for industry like the University of Heidelberg (simulation of flame instabilities) or the Central Institute for Applied Mathematics (simulation of granular materials).

Realizing that there is a strong relation between the use of high performance computers in a country and its potential in high technology, the small portion of industrial users can not be accepted for the future. In nearly every kind of industry computer simulation becomes more and more important. With simulation complexity and corresponding calculation requirements increasing daily, the use of the most powerful computers is a conditio-sine-qua-non. The architectural concept of massive-parallelism is not the easiest architecture for the programmer, but it is the most powerful — and the distance to conventional computers is increasing.

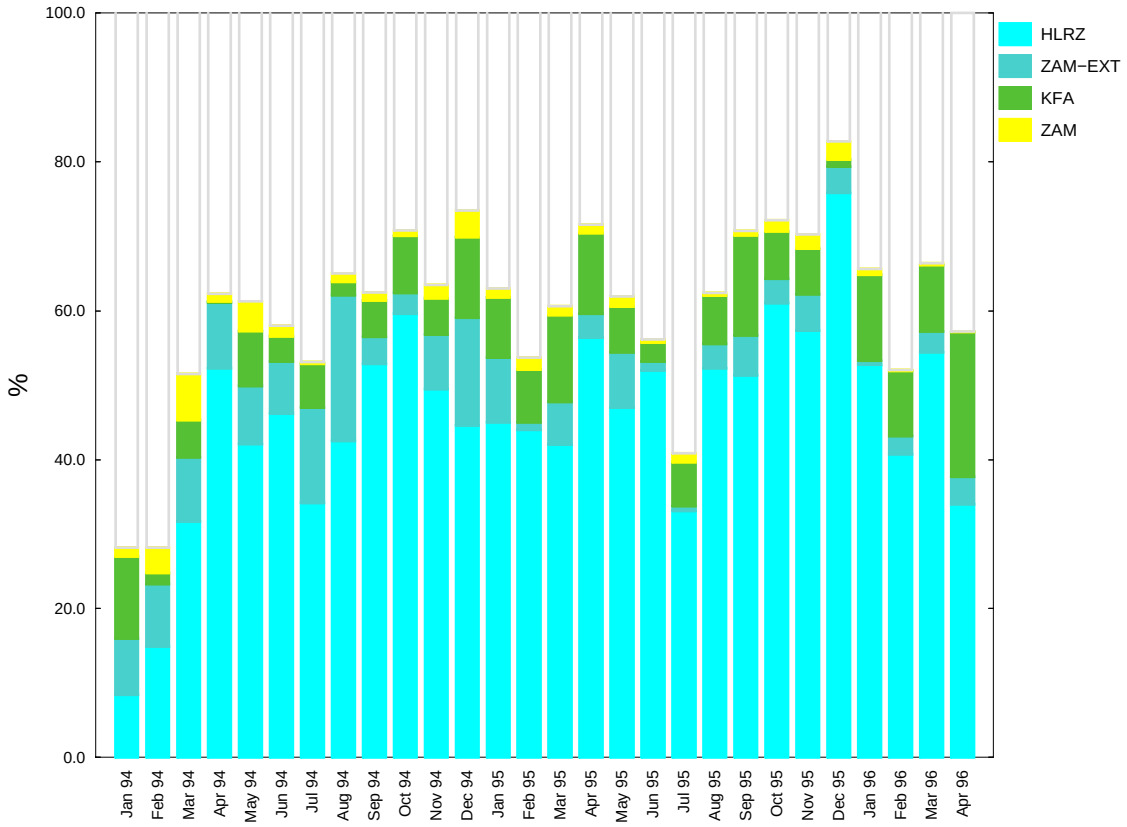


Figure 4: Users of Intel Paragon XP/S-10 at ZAM

In 1996, new computers will be installed at the Research Centre Jülich, a CRAY T90, a CRAY T3E and a CRAY J90. A significant part of their compute capacity will be offered to industry. Looking at the first years of massively-parallel computers, which were characterized by many usage problems, it is not astonishing that commercial and industrial users were not enthusiastic about this architectures. In the meantime, the situation has improved significantly. The community of computational science & engineering is now requested to demonstrate the usefulness of this novel technology to all relevant groups outside. This report is intended to work in that direction.

2. List of applications

In the following all applications have been listed, which have applied for compute time on the Intel Paragon XP/S since its installation at the Research Centre Jülich in 1993[1]. There is certainly some overlap in the different subject areas, but it was decided not to eliminate them here to reflect also the quantitative distribution of users among the different fields of applications. Classifying certain applications as either scientific or industrial may not be completely correct from some point of view; the structuring introduced is only meant as a general outline.

2.1 Mathematics

- Adaptive techniques in the Multisplitting Waveform Relaxation algorithm (University of Queensland)
- Advanced Monte Carlo algorithms (University of Regensburg)
- Analysis of the parallel equation solver Paragon Prosolver DES (Research Centre Jülich)
- Diagonalisation of large Non-Hermitesian matrices (Research Centre Jülich)
- Discrete logarithms (University of Saarland)
- Development of parallel software for numerical applications (University of Karlsruhe)
- Mathematical Software (Research Centre Jülich)
- Matrix diagonalisation (Research Centre Jülich)
- Monte Carlo simulation (University of Cologne)
- Monte Carlo simulation of systems (University of Duisburg)
- Parallel algorithms for numerical linear algebra (University of Wuppertal)
- Parallel equation solvers (Research Centre Jülich)
- Partial differential equations (GMD)
- Stochastical optimization (Technical University of Chemnitz-Zwickau)
- Distributed algorithms for sparse matrices (ZIB)
- Wavelet analysis (University of Frankfurt)
- Two-dimensional Riemann problems for the Euler equations (ETH Zurich)

2.2 Physics

- Two-dimensional Ising models (University of Cologne)
- Aging in two-dimensional Ising Spin Glasses (University of Cologne)
- Computation of phase diagrams of the Duffing and van der Pol equations (Research Centre Jülich)
- Computation of Skyrmion dynamics on a space-time-grid (Research Centre Jülich)
- Calculation of Splash Function for sand grains (Research Centre Jülich)
- Computational Fluid Dynamics (ETH Zurich)
- Simulation of molecular beam epitaxy (Research Centre Jülich)
- Domain growth in 3D Ising Spin Glasses (University of Cologne)
- Dynamics of molecular beam epitaxy (University of Cologne)

[1] There are some cases, where institutes applied for compute time but did not use it later.

- Dynamics of interacting charges (University of Marburg)
- Dynamic mass generation (Research Centre Jülich)
- Influence of process rates in one-dimensional reaction-diffusion-systems (University of Bonn)
- Electron structure of solid matter (Research Centre Jülich)
- Electronic structures for ideal crystals with the KKR-method (Research Centre Jülich)
- Energy loss of heavy ions in dense plasms (Technical University of Darmstadt)
- Development of Fit programs (University of Bonn)
- Evolution equations for two-dimensional current density distribution (University of Münster)
- Flow tube profiles in Hamiltonian lattice gauge theory (GSI)
- Spin systems and lattice theory (University of Mainz)
- Gluon Propagator (DESY)
- Correlation functions in non-linear systems (Research Centre Jülich)
- Correlation length and border area tension for Potts models (University of Mainz)
- Critical behaviour of thinned anti-ferro magnets (University of Duisburg)
- Likelihood-method for matching dynamic models with experimentally generated data (Research Centre Jülich)
- Magnetic properties of non-periodic systems (University of Halle-Wittenberg)
- MIT in the Anderson-Mott-Hubbard-model (Technical University of Braunschweig)
- Modelling of crystal growth using Czochralski's method (Research Centre Jülich)
- Molecular dynamics of granular media (ESPCI)
- Molecular dynamic simulations of martensitic phase transforms in iron-dominated alloys (Univesity of Duisburg)
- Molecular dynamic simulation of synthetic polymeres (MPI for Computer Science)
- Molecular dynamic simulation of the non-local dielectric function of water (Research Centre Jülich)
- Molecular dynamics of granular media (Research Centre Jülich)
- Monte Carlo simulation of bipolar systems (Research Centre Jülich)
- Monte Carlo simulation of percolation and conductivity of three-dimensional grids (Research Centre Jülich)
- Monte Carlo simulations in statistical physics (University of Cologne)
- Monte Carlo simulations of epitaxy (Research Centre Jülich)
- Monte Carlo statistics of CH molecule impacts on (111)-diamond surfaces (Technical University of Chemnitz-Zwickau)
- Neural networks for generalization behaviour of multi-layer networks (University of Regensburg)
- Numerical integration of the transsonic flow around a cylindre (ETH Zurich)
- Surface diffusion in systems with phase transitions (Research Centre Jülich)
- On the structure of parameter space of laser models (Research Centre Jülich)
- Optimizing structures by using genetic algorithms (Research Centre Jülich)
- Parallel particle simulation of low-temperature plasms (University of Bochum)
- Parallel simulation of turbulent flow (University of Cologne)
- Parallelization of the Monte Carlo code EIRENE (Research Centre Jülich)
- Parallelization of algorithms for particle simulation (University of Bochum)
- Parallelization of cluster algorithms (University of Regensburg)
- Parallelization of CFD solvers (University of Freiburg)
- Phonons in strong correlated electron systems on finite grids (University of Bayreuth)

- Potentials in SU(2) lattice gauge theory (DESY)
- Quantum chromo dynamics at high temperatures (University of Bielefeld)
- Quantal analysis of simulated EPSCs (University of Göttingen)
- Quark anti-quark potential on asymmetric grids (DESY)
- Schrödinger Poisson system (IAAS)
- Iteraction of heavy ion beams with a plasm (Technical University of Darmstadt)
- Self averaging in critical disordered systems (Weizmann Institute of Science)
- Simplical quantum gravity and random surfaces (University of Bielefeld)
- Simulation of gas discharges (University of Kaiserslautern)
- Simulation of multi phase flow (ESPCI)
- Simulation of elementary particle distribution in the phase space (Research Centre Jülich)
- Simulation of phase transitions in liquid crystal systems (University of Dortmund)
- Simulation of polymer systems (University of Heidelberg)
- Simulation of systems with magnetic volume instabilities (University of Duisburg)
- Simulation of two-phase systems (Research Centre Jülich)
- Simulation of cellular structures (Research Centre Jülich)
- Many particle systems (Glasgow University)
- Computational fluid dynamics with branches (University of Erlangen)
- Structure development phenomena in growth processes (Research Centre Jülich)
- SU(3) Theta Vakua (Research Centre Jülich)
- Susceptibilities and cumulants in two-flavour QCD (University of Bielefeld)
- System diagnosis and prediction (University of Bochum)
- Theoretical modelling of surface-induced ordering (University of Mainz)
- Theta vakua in SU(2) lattice gauge theory (DESY)
- Tree-codes in molecular dynamics simulations (University of Erlangen)
- Disorder by thinning in frustrated anti-ferro magnets (University of Göttingen)
- Analysis of non-linear fall equations (University of Ulm)
- Analysis of phase transitions by influence of random fields (University of Mainz)
- Many particle systems (Research Centre Jülich)

2.3 Biology, chemistry, medicine

- Spreading of trace substances in the atmosphere (Research Centre Karlsruhe)
- Medical Imaging (Research Centre Jülich)
- Biomagnetic data analysis (Research Centre Jülich)
- Correlations in biological neural networks (University of Ulm)
- Numerical simulation of biological systems: The retina (University of Oldenburg)
- Optimizing strategies in biological computing (University of Bochum)
- Simulation of solid oxide fuel cells (Research Centre Jülich)
- Reconstruction of micro-circulation networks (Technical University of Aachen)
- Spread of harmful chemicals in the soil (Research Centre Jülich)
- Structure refinement of bio molecules with molecular dynamics (University of Frankfurt)
- Ab-initio molecular dynamics (FU Berlin)
- Attributes of metal clusters (FU Berlin)

2.4 Astronomy, meteorology, geology

- Simulating pulsar magnetospheres on parallel computers (University of Tübingen)
- Simulation of nipple firmation on the surface of the planet Mars (Research Centre Jülich)
- Beam-hydrodynamic simulation of the development of starburst regions (University of Kiel)
- Parallelization of a mesoscalic meteorological model [EURAD] (University of Cologne)
- Parallelization of the IFS weather code of ECMWF (GMD)
- Dynamics of ditch networks of tectonic plates (Research Centre Jülich)
- Simulation of a two-dimensional spring-block model for earth quakes (Research Centre Jülich)

2.5 Computer science

- Efficiency of parallel processing (University of Magdeburg)
- Evolution strategies and analysis of chaotic systems (University of Bochum)
- Genetic algorithms and neural networks (Research Centre Jülich)
- Gossiping (all-to-all total-exchange) on grid computers (MPI for Computer Science)
- SVM-Fortran (Research Centre Jülich)
- Implementing a parallel programming language with implicit parallelism [EFECT] (HMI)
- Metacomputing (Research Centre Jülich)
- Implementing a dynamic load balancing system (University of Stuttgart)
- PARvis/VAMPIR (Research Centre Jülich)
- Metacomputing (Paderborn Center for Parallel Computing)
- Genuine parallelizing (Research Centre Jülich)
- Metacomputing (GMD)
- Shared Virtual Memory (Technical University of Munich)
- Visualisation of multi-dimensional arrays (Research Centre Jülich)
- Parallelization of the chess program ZUGZWANG (University of Paderborn)

2.6 Industrial applications

- Benchmarking new architectures (GENIAS Software)
- Computations on the Niob Saphire border area (MPI for metal research)
- Properties of self learning steerings (Technical University of Chemnitz-Zwickau)
- Instabilities and soot generation in flames (University of Stuttgart)
- Investigation of elastic disc systems (Research Centre Jülich)
- Numerical model computations for the development of micro structures (MPI for metal research)

- Optimizing constellations of low flying communication satellites (Technical University of Braunschweig)
- Parallel simulation of flame instabilities (University of Heidelberg)
- Simulation of granular materials (Research Centre Jülich)
- Simulation of fracture processes in composite materials (Research Centre Jülich)
- Computational fluid dynamics (Technical University of Aachen)
- Structural dynamics (Intes GmbH)
- Structural dynamics (Condat GmbH)
- Structural mechanics and temperature field computation with SMART (Research Centre Jülich)
- Failing problems in structural mechanics (University of Karlsruhe)
- Traffic simulation (University of Cologne)
- Fast simulation systems for integrative traffic analysis and steering (University of Cologne)
- Simulation of multi lane traffic flow (University of Cologne)

2.7 Compilers, tools, and maintenance of parallel computers

- Development of the Shared-Memory-System SVM (Intel)
- Extending of the MACH-based MAX-SVM-System (Technical University of Munich)
- Evaluation of HPF (ETH Zurich)
- Fortran90 (Research Centre Jülich)
- Optimizing of applications (Intel)
- Parallel I/O for HPF-compilers (University of Vienna)
- Performance analysis (Research Centre Jülich)
- PARMACS (PALLAS)
- Parallel processing training course (University of Heidelberg)
- System management tools (Technical University of Aachen)
- System-Software-Tests (Intel)
- Test of the accounting system MACS (Intel)
- TOP2 (CRAY Research)
- Tools (GMD)
- Porting applications with ADAPTOR (GMD)

3. Application reports

As one can see in the chapter before there are many applications, too many to present all of them in more detail. Thus, in the following we have collected some significant applications covering all fields of applications, which shall give a sufficient overview on the whole spectrum and sufficient insight in problems which are fitted for
massively parallel computers today.

Groundwater flow simulations

O. Neuendorf

Research Centre Jülich, Institute of Petroleum and Organic Chemistry (ICG-4)

Project: Behaviour of pollutants in geological systems

Director: H. Vereecken

Many of the current environmental problems are caused by the transport of potentially hazardous agricultural and industrial chemicals in soils and aquifers. Mathematical models are increasingly being used to quantify the transport of these pollutants, to establish risk assessments, and to design and evaluate possible sanitation strategies. A common approach to describe the transport processes in soils and aquifers has been to model water and solute transport in a deterministic way using partial differential equations, derived from basic physical and chemical principles. It has been recognised that due to the inherent spatial variability of soil and aquifer parameters, this traditional approach may be limited for field and large scale problems. To account for the effect of spatial variability on the transport processes, stochastic models have been developed. These models rely on stochastic partial differential equations in which the hydraulic properties are viewed as realisations of space random functions. To preserve the given statistical structure of these properties, the size of the numerical grid must be large (several nodes per correlation length) and the flow domain has to be extended over several correlation scales in three dimensions. In addition these transport processes may be strongly non-linear (e.g. water flow in the vadose zone) and time dependent. Former considerations result in large equation systems which presently can only be handled on either vector or parallel computers in combination with appropriate numerical algorithms.

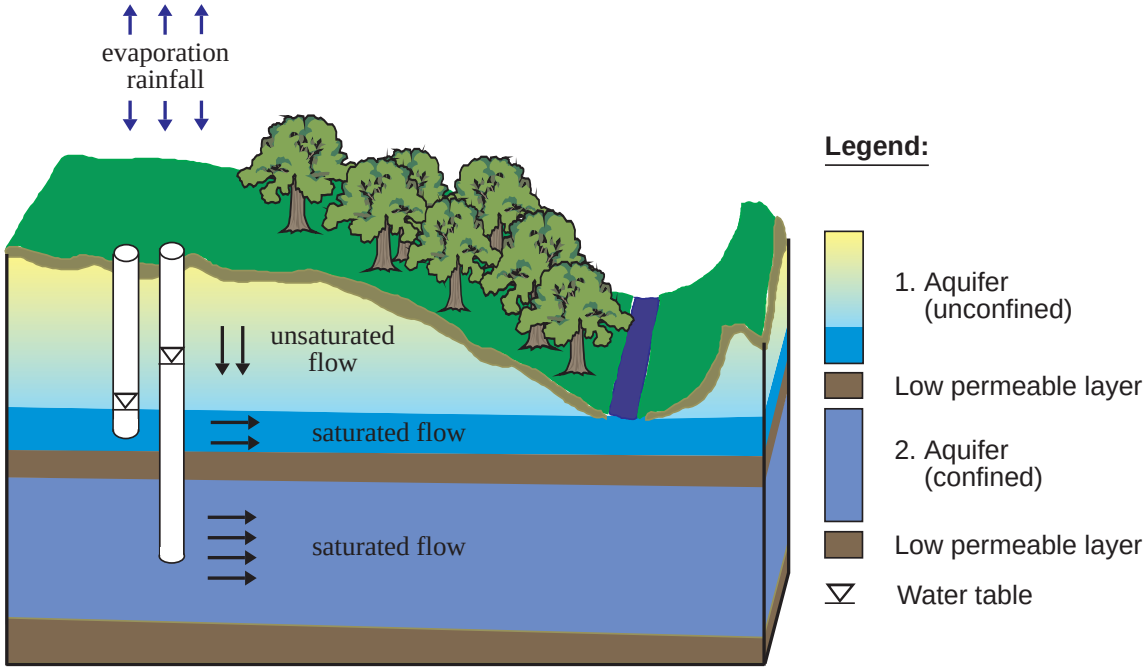


Figure 1: Model for the simulation of the groundwater flow

The program TRACE was developed to simulate the water flow and solute transport on massively parallel systems like the Intel Paragon. It is capable to deal with the order of up to

106 elements including heterogeneous non linear parameters. It is based upon a finite element algorithm with a Picard method for the non linear equation system and a conjugent gradient method for the linear one. For the solution of the solute transport an additional particle tracking algorithm was developed and incorporated in the program PARTRACE. For the parallelization of the finite element algorithm the Schwarz domain decomposition method was chosen. This method can be easily incorporated in existing models and used on either vector computers with several processors or massively parallel computer systems. Figure 2 gives an example for the performance of the Schwarz domain decomposition technique.

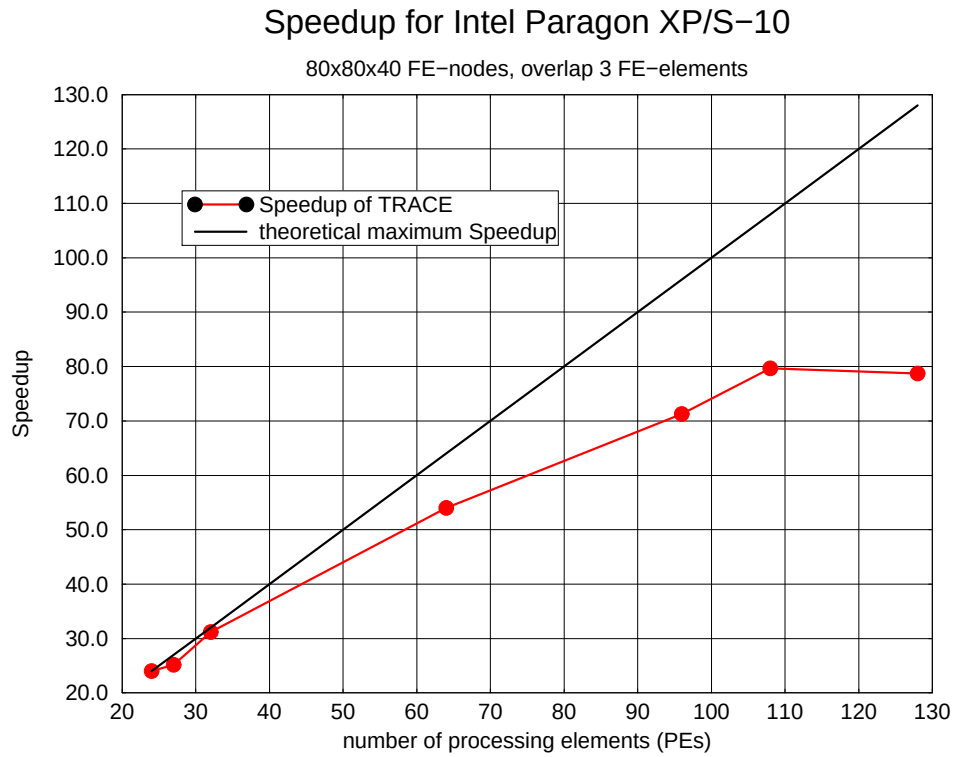


Figure 2: Speedup of the groundwater simulation on Intel Paragon XP/S-10

References

1. Vereecken H., Lindenmayr G., Neuendorf O., Döring U., Seidemann R., (1994), TRACE A mathematical model for reactive transport in 3D variably saturated porous media. Internal Report KFA-ICG4-501494, Jülich.
2. Vereecken H., Neuendorf O., Lindenmayr G., Basermann A., (1995), A parallel Schwarz domain decomposition method for the numerical solution of transient water flow in heterogeneous porous media. Internal report KFA-ICG4-500795, Jülich.

Distinct element simulations

G.A. Kohring

Research Centre Jülich, Central Institute for Applied Mathematics

Project: Mathematical Methods and Algorithms for New Computer Structures

Director: P. Weidner

Distinct element simulations are based upon the use of distinct, individual elements each of which is free to move and interact with the other elements in accordance with some a priori defined rules. This is quite different from the continuum approach customarily used in science and engineering applications, where only a small number of differential equations govern the behaviour of the entire system. The first simulations dealt almost exclusively with spherical molecules interacting via simple molecular potentials, hence the widespread use of the term „molecular dynamics“ for describing these methods. However, modern computers have made it possible to relax the condition of spherical elements and to use quite complex interaction models.

One area where distinct element simulations are enabling scientists and engineers to probe phenomena which still are not well understood, is the industrially important field of granular materials. Granular materials are found throughout the world: examples range from edible grains to the powders used for making turbine blades. Granular materials pose a difficult problem for distinct element simulations because the grains can be of arbitrary shape and can experience many different types of interactions. Typical interactions include visco-elastic repulsion, surface adhesion and surface friction.

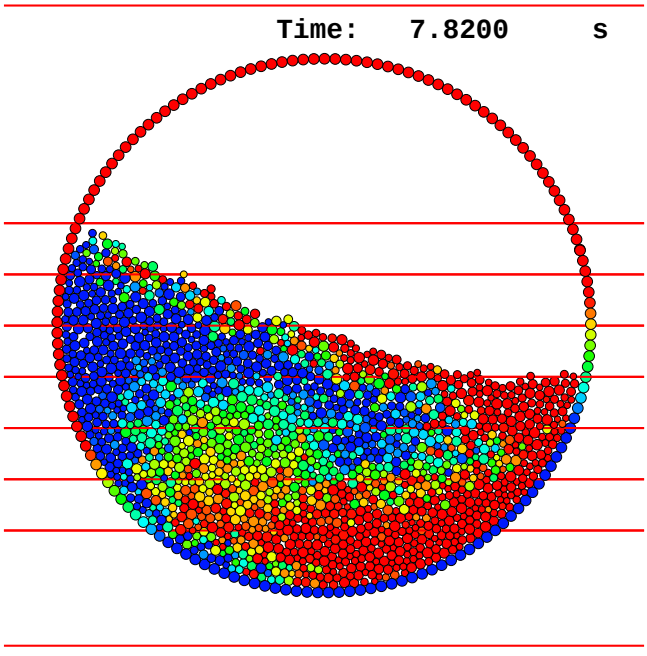


Figure 1: A simple drum used for mixing different granular materials.

The advent of parallel computers promises to revolutionize this field, because of the inherent parallelism in all of the algorithms used for distinct element simulations. Basically, each algorithm depends upon performing the same set of calculations for each element in the system. Parallelization techniques for distinct element algorithms depend upon the range of the interactions. For granular materials, the interactions are short ranged and the link cell approach, a form of domain parallelization, is the most efficient method. This method divides the physical space into small cells and assigns each element to a given cell. If the cell size is larger than the element's interaction radius, then only the neighbouring cells need be checked in order to find all possible collision partners. One of the most difficult problems in simulating the dynamic behaviour of granular materials on parallel computers, is obtaining a proper load balance across all processors. This problem can be overcome by mapping the domains to the processors dynamically. A local procedure for redistributing the particles is also essential in order to avoid the complexity inherent in global allocation schemes.

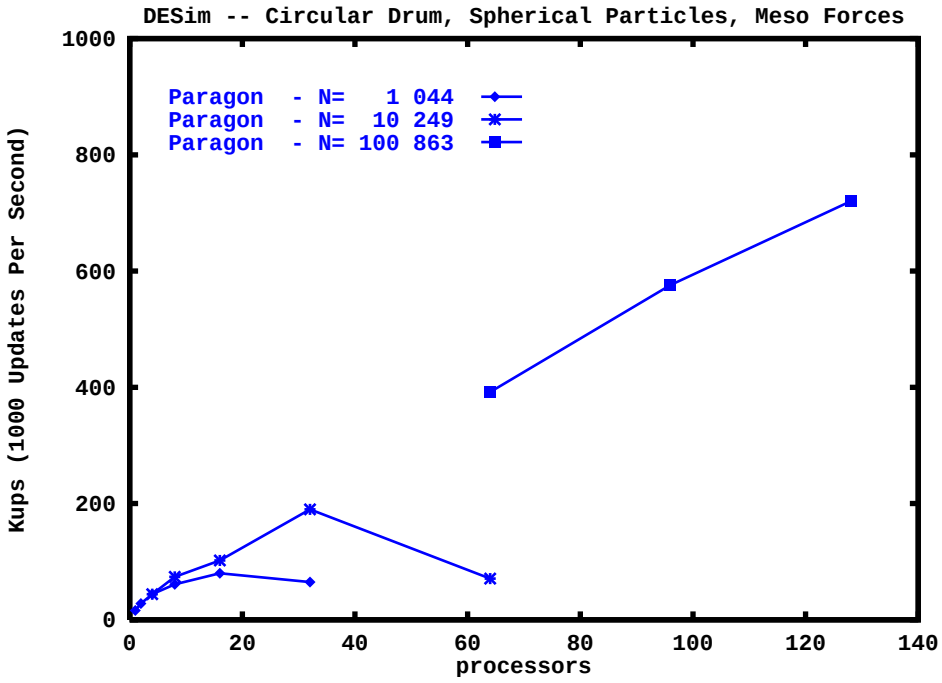


Figure 2: Performance of the parallel algorithm on the Intel Paragon XP/S-10.

References

1. Kohring, G.A.: Studies of Diffusional Mixing in Rotating Drums Via Computer Simulations, to appear in: Journal de Physique I, 1995
2. Knecht, R; Kohring, G.A.: Dynamic Load Balancing for the Simulation of Granular Materials, Proceedings of ICS´95, Barcelona, July 1995, pp. 164-169

A parallel industrial code for deformations with contact

B. Saddak, D. Vinckier

Research Centre Jülich, Central Institute for Applied Mathematics; Condat GmbH

Project: CONDAT-DYNA3D (funded by BMBF)

Director: P. Weidner, J. Kiermeir

Finite element programs using an explicit time integration scheme for nonlinear structural dynamics are rapidly gaining interest. Applications for this class of programs range from simulations of quasistatic material tests over crash and metal forming simulations up to the analysis of high velocity phenomena involving extensive damage and material erosion. CONDAT uses its finite element program CONDAT-DYNA3D in analysis projects for European companies and government organisations in all domains mentioned above [1][2]. Due to the high number of elements (typically 50000 to 600000 degrees of freedom) and the very small time step (typically 0.01 to 5u), computing times can be very high in industrial applications. To investigate the feasability of an efficient and general, i.e., machine independent parallelization of these so-called "hydrocodes", a research project, funded by the German BMBF, was initiated to develop a machine independent parallel version of CONDAT-DYNA3D. The performance of the parallel version and the potential for future developments is evaluated on different parallel architectures.

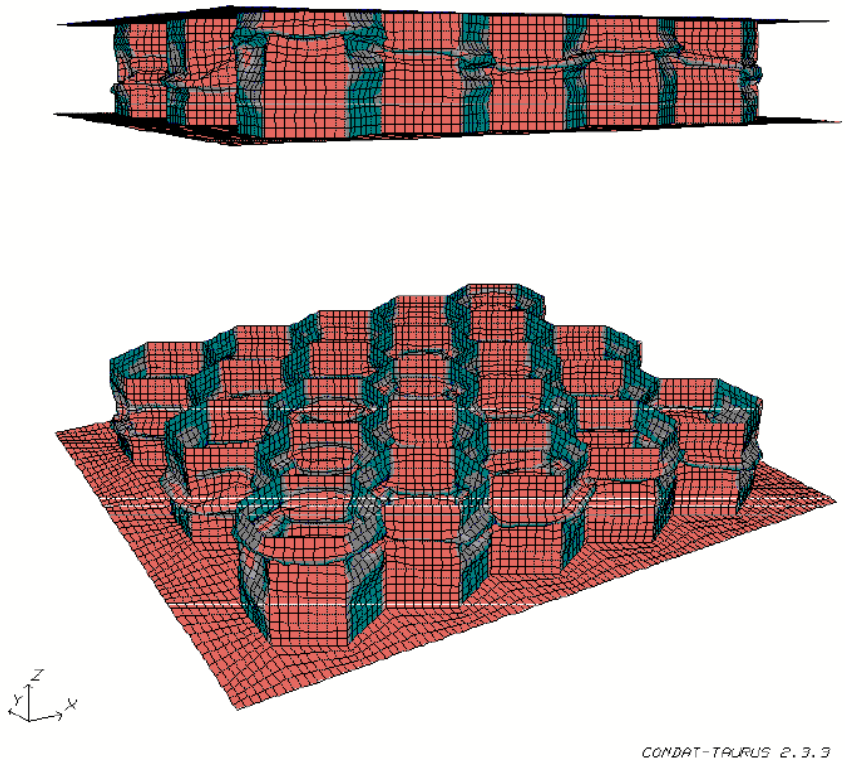


Figure 1. Honeycomb structure crash

The parallel version of CONDAT-DYNA3D which contains all important features including the contact module, is based on a domain decomposition method and applies message passing techniques. The domain decomposition uses the *Multilevel Recursive Spectral Bisection method* (MRSB) that achieves a high-quality decomposition at low computational costs. In our

application the domain decomposition is done once at the beginning and is calculated sequentially. Here, the partitioning time is almost negligible compared with the run time of the simulation program. Thus a parallelization of the decomposition tool is not considered.

Prelimary tests with CONDAT-DYNA3D benchmarks showed that for models with less than 10000 elements good speedups can be reached with up to 16 processors. Beyond this number, the computing efficiency drops as the domains become too small, and the overhead for communication and model management increases. The crushing of an aluminium honeycomb structure is chosen as an industrial benchmark for the parallel version of CONDAT-DYNA3D. Honeycomb structures are used inside impact barriers for car crash experiments. Simulations are very useful in the process of understanding the crushing characteristics and developing equivalent barrier models to be used in car crash analysis. The simulation model, a brick like structure of 7 x 7 honeycomb cells, contains 59648 elements and is crushed in the direction of the cell axis. Figure 1 shows the crushed structure developing a charcteristic shear band. The speedup factors obtained on the Paragon XP/S 10 of the Research Centre Jülich with up to 138 processors are near the optimum as shown in figure 2. Here the deformation was simulated until the first contact occurred. The simulation with contact for this structure is ongoing work. The performance of simulations involving extensive and rapidly changing contacts still needs to be improved. The integration of the charcteristics of contacts in the decomposition algorithm and the realisation of a dynamic decomposition will result in a further gain in performance and flexibility of the program on high performance parallel computers.

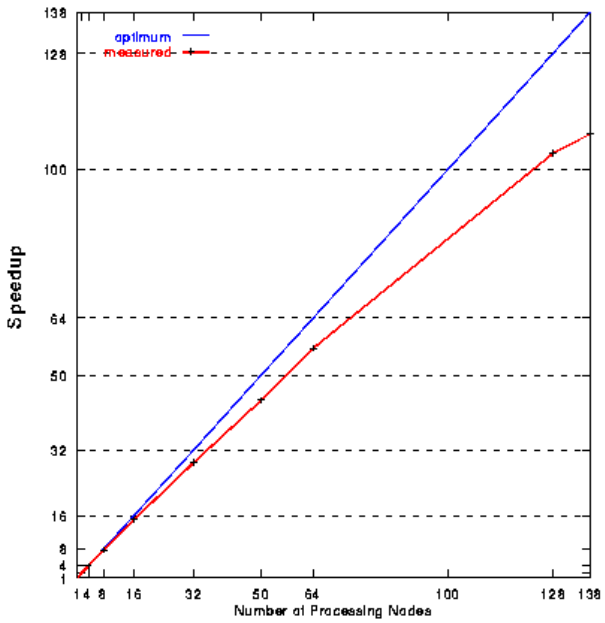


Figure 2. Speedup of honeycomb structure crash on Intel Paragon XP/S-10

References

1. Maier, V. Altstädt, D. Vinckier, K. Thoma, Numerical and experimental analysis of the compact tension test for a group of modified epoxy resins, Journal of Polymer Testing, 13 (1994), pp. 55-66
2. Thoma, D. Vinckier, Numerical analysis of a high velocity impact of fiber reinforced materials, IMPACT IV, Post-Conference Seminar of the 12th Int. Conference on Structural Mechanics in Reactor Technology (SMiRT), (1993)

Parallelization of the Finite Element code SMART-convection

Astrid Watermann

Research Centre Jülich, Institute for Safety Research and Reactor Technology

Project: Structural Mechanics in Safety Analysis

Director: J. Altes

Massively parallel computers can be successfully used to shorten calculation times of numerical applications, e.g. the solution of systems of partial differential equations based upon the finite element method. However, problems arise when the power of these parallel machines is to be used for calculations with large program systems which were, in first stage, developed for serial architectures. These programs employ the respective program and data structures which often result in poor speedup and efficiency. However, those codes used in many institutions cannot be substituted by new parallel ones in near term range, because of costs, their size and experience with this classical codes. Therefore, a parallelization strategy for the finite element code SMART has been developed and implemented on Intel Paragon. In general a finite element calculation can be divided into three parts: (i) calculation of element matrices, (ii) assemblage of these matrices to the system matrix, and (iii) solution of a system of linear or non-linear equations. The concept of parallelization considers, beside these three principal parts, in addition data management and I/O. In order to get an overall satisfactory speedup, a kind of data parallel strategy has been introduced.

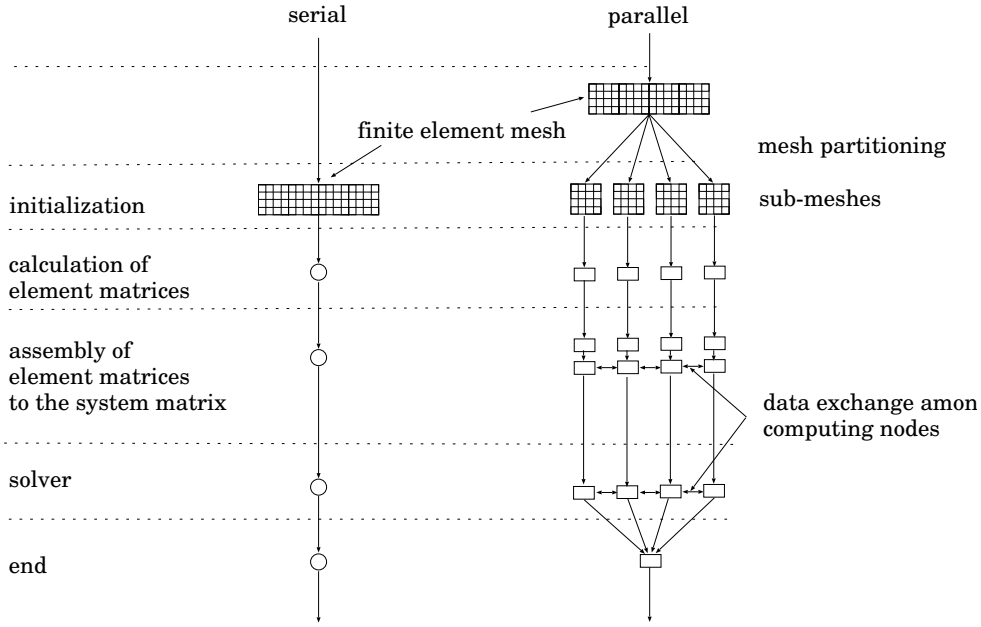


Figure 1: Overview of global parallelization strategie

Regarding parallel computers with distributed memory such as the Intel Paragon, the most important feature of the strategy is a geometric mesh partitioning, which assigns disjunct clusters of finite elements to the processors of the parallel machine [2]. Assuming equally distributed computing load and data for the local processors or meshes, linear speedup is only reduced by communication times. After such a partitioning, calculation starts on each processor with a serial calculation of SMART, but only for the locally stored elements. Figure 1 depicts the concept of such a calculation. While computation of the element matrices is performed locally on each processor for the managed elements, assemblage of the system matrix has to be done taking account of the dependencies among the sub-meshes on different

processors. Therefore the assembly phase is divided into two parts. In the first part, assemblage is only done for the locally stored elements while in the second part data are exchanged between different processors. This is the first step in the SMART calculation where communication is necessary. For this step, a communication scheme has been developed, which enables asynchronous exchange of data, i.e. non-zero elements of locally stored matrix rows. The advantage of this scheme is that communication is being done on matrix level. The programmer, therefore, simply needs to know the storage scheme of the matrices, while other parts of the internal data management of the finite element code are not relevant. After this step, the global system matrix is stored row-wise and distributed to the parallel processors. An efficiently parallel executable solution algorithm for the systems of equations is then introduced. For the code SMART, the direct Cholesky method has been replaced by the iterative conjugate gradient algorithm [1], which shows good performance and is easy to implement with respect to the given storage scheme of the system matrix.

However, some additional changes are required for the output of the solution vector, which has to be assembled among the distributed memories of the parallel processors. Furthermore, some optimization for the considered architectures may be necessary, i.e. I/O regarding Intel Paragon. Speedup is demonstrated for the example of the Rayleigh-Benard convection regarding a finite element grid with 10,000 elements. For calculation of an instationary convection-dominated flow in each time step, the temperature field with about 10,000 unknowns and the velocity field with about 20,000 unknowns have to be solved using the finite element method.

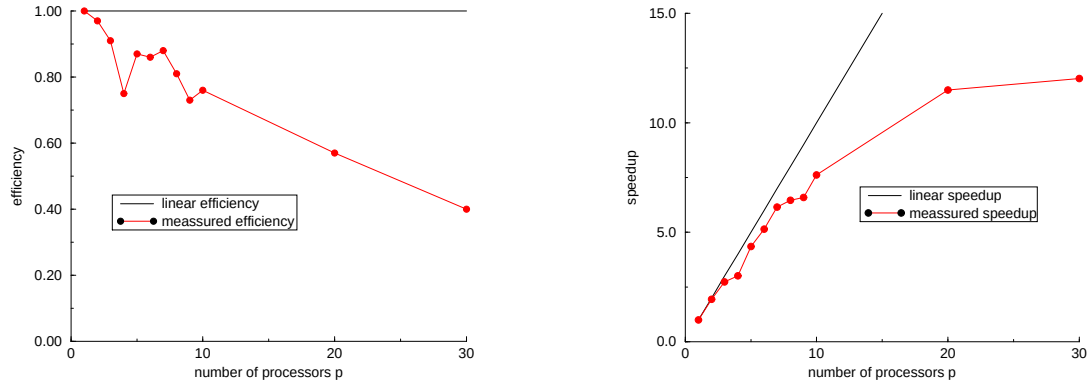


Figure2: Speed up and efficiency for solver

Figure 2 shows speedup and efficiency for one time step including CPU, system, I/O, communication, and idle times. It shows a good speedup on up to ten processors and, even when employing 20 processors, computing times can be reduced. In contrast, the use of more than 20 processors does not speed up the calculation any further, which becomes obvious when looking at the small problem size.

References:

1. L.G.C. Crone, The Preconditioned Conjugate Gradient Method on Distributed Memory Systems, *High Performance Computing and Networking*, Springer, 1994
2. C. Farhat, A Simple and Efficient Automatic FEM Domain Decomposer, *Computers & Structures* **28**(5): 579-602, 1988
3. A. Watermann, Parallelisierung des Finite-Element-Codes SMART-Konvektion, Jül-Bericht 3122, ISSN 0944-2952, Institut für Sicherheitsforschung und Reaktortechnik, Forschungszentrum Jülich GmbH, 1994

A tool-supported parallelization of Finite Element applications

Björn Reichel

Research Centre Jülich, Central Institute for Applied Mathematics

Project: PARFEM (funded by BMBF)

Director: K. Solchenbach, PALLAS GmbH

One of the important questions is how to solve the large, sparse, symmetric and positive definite system of linear equations from finite-element discretization. The Central Institute for Applied Mathematics at Research Centre Juelich considers the method of conjugate gradients (CG). In particular for very ill-conditioned matrices, sophisticated preconditioners are necessary to obtain both acceptable convergence and accuracy of CG [1][2]. Variants of incomplete Cholesky preconditioniers are shown to be well suited for massively parallel maschines. The algorithm of preconditioned CG is implemented in a modified variant with better parallel properties. For the incomplete Cholesky preconditioner, we consider quadratic diagonal blocks from the original matrix A. If we distribute the matrix A in block-rows to the processors, the complete work for the preconditioning can be performed without communication, that is determining the preconditioner and solving the preconditioning system in each iteration.

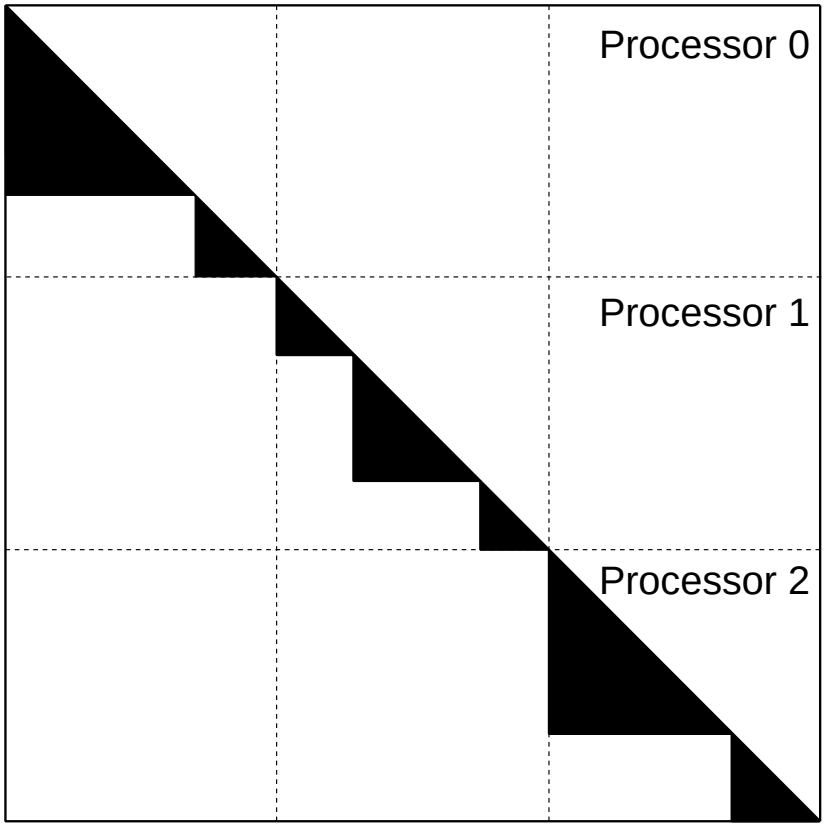


Figure 1: Block-decomposition of the local block for preconditioning

The total number of elements in the preconditioner depends on two parameters. The number of elements in the lower triangular system where the forward/back-substitution has the best MFLOPS rate on the computer system used is the machine dependant parameter. The

percentage of the non-zeros in the skyline of the local diagonal-block per processor is the problem dependant parameter. If the non-zeros of the local block are closely packed in its skyline, we assume that a lot of information is stored close to the diagonal of the matrix. Otherwise, if the non-zero structure is very sparse, we assume that most information is outside the local blocks. The product of the two parameters gives the basic value for the number of elements in a lower triangular subblock of a processor's local diagonal-block.

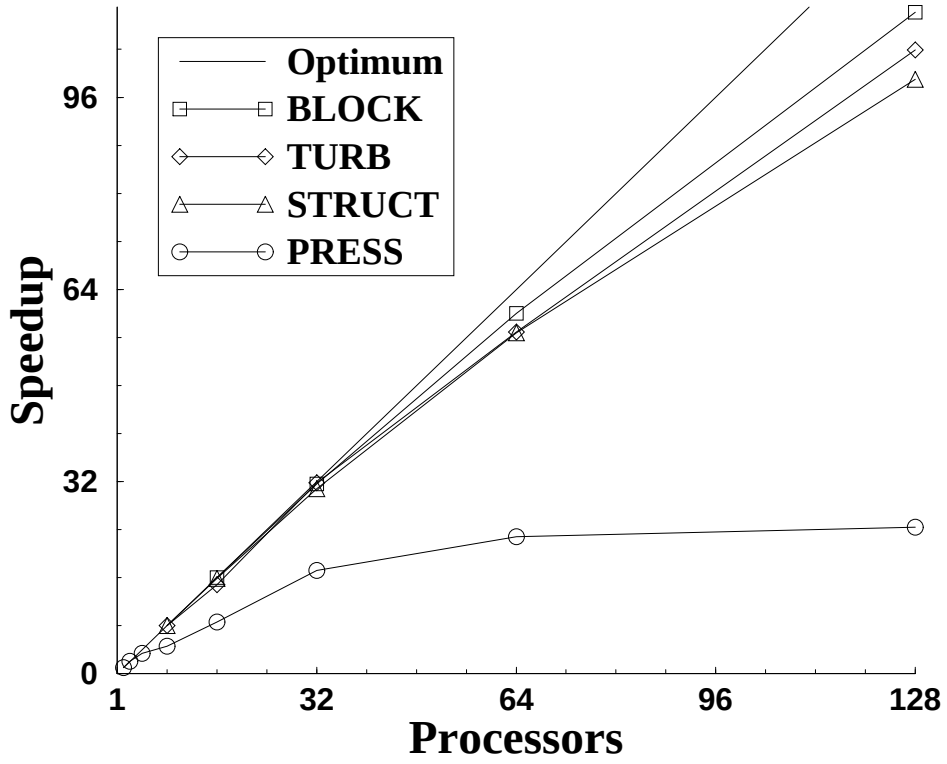


Figure 2: Speedup of the iterative PERMAS solver for different benchmarks

For large problems as BLOCK, TURB and STRUCT high speedups up to 110 on 128 processors are achieved. For these problems the number of iterations even decreases slightly with increasing the processor number, because the non-zero structure close to the diagonal is compact. Hence the local blocks become smaller but also denser with increasing processor number, and thus more elements are considered for the preconditioner. This improves the preconditioning effect. The matrix PRESS is a small one, also with a compact non-zero structure around the diagonal. The preconditioning effect decreases markedly since the local blocks become too small for high processor numbers, and thus considerably less elements are taken for the preconditioner. This causes the worse speedup behaviour.

References:

1. O.Axelsson, Iterative Solution Methods, Cambridge University Press, 1994
2. R.Barrett, M.Berry, T.Chan, J.Demmel, J.Donato, J.Dongarra, V.Eijkhout, R.Pozo, C.Romine, and H.van der Vorst, Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, SIAM, Philadelphia, 1993

Parallelization strategies for equations of balance in an atmospheric model

Andreas Müller

University of Stuttgart, Institute of Parallel and Distributed High-Performance Systems (IPVR)

Project: Development of Meteorological Computational Models (Univ. of Karlsruhe)

Director: F. Fiedler

The equations of balance describing the fields of concentrations of different chemical reactants in the atmosphere form a system of nonlinear partial differential equations in space, time and the considered species.The numerical solution is calculated by splitting the participated processes. The physical processes are modelled by the program package DRAIS and the chemistry by RADM. All solvers use a common three dimensional grid, which is embedded in the considered physical area. Because of the involvement of different timescales in the processes, different numerical methods are used. In DRAIS for each species three one dimensional equations of advection and of diffusion are solved one after the other. The equations of advection are solved by FCT-algorithms (flux corrected transport) and the diffusion by a standard difference scheme. RADM solves the stiff system of ordinary differential equations with a modified predictor-corrector scheme. Simulations in scientific computing often consist of the simultaneous treatment of different physical phenomena described by systems of partial nonlinear differential equations.

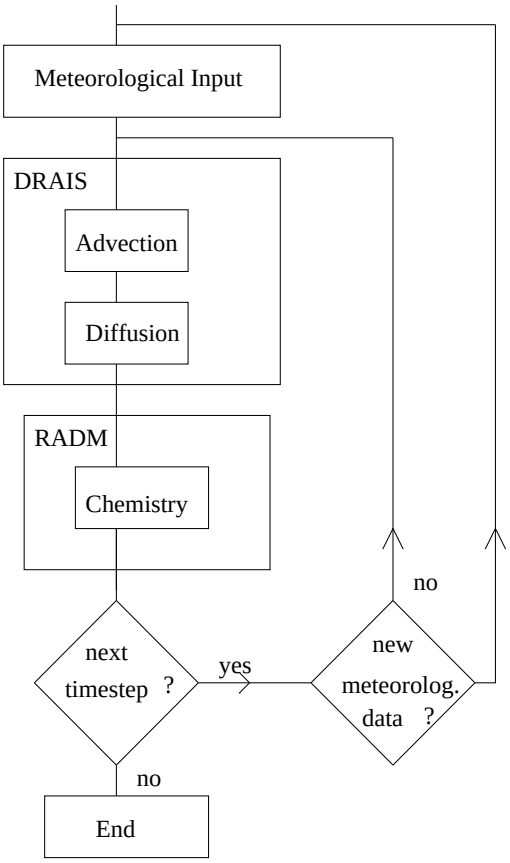


Figure 1: Flow chart

The parallelization of DRAIS and RADM has at the one hand been realised by domain decomposition of the considered space and at the other hand by seperating the calculation of the different species on different processors. This led to the use of different numbers of coordinates concerning the domain decomposition between the parallelization of DRAIS and of RADM. Because of the independence of the calculation of different species the high expense of communication between DRAIS and RADM could be hidden behind the computation by the use of asynchronous communication.

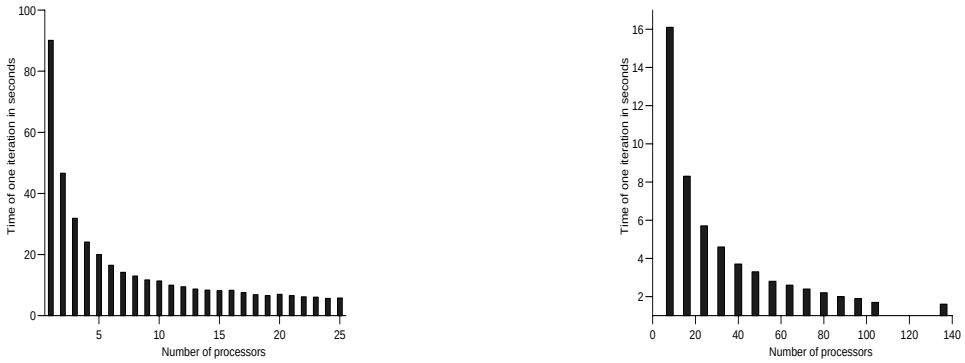


Figure2: Iteration times for one iteration

Figure 2 gives measurements of the accelerations of the scalable program by using different numbers of processors without refering to the special strategy. The main loss of efficiency is caused by the different expense between different grid points in the chemistry. By the used fixed size of the model (about 60.000 grid points) the grid points with higher expense of the computing time are filtered out by an increasing number of engaged processors and cause a strong reduction of the efficiency. Further reasons for the loss of efficiency are a worse use of special compiler features (pipelining tools) and a worse load balance because of the properties by division of integers by an increasing number of engaged processors.

Acknowledgement. This work has been done by the author at the Institut für Meteorologie und Klimaforschung of the Research Centre Karlsruhe and of the University of Karlsruhe. The Research Centre Jülich I thank for the possibility of using the Paragon XP/S.

References

1. J.S. Chang, R.A. Brost, I.S.A. Isaksen, S. Madronich, P. Middleton, W.R. Stockwell, C.J. Walcek, A Three-Dimensional Eulerian Acid Deposition Model: Physical Concepts and Formulation, J.Geophys. Res. 92, 14681 - 14700, 1987
2. F. Fiedler, Development of Meteorological Computer Models, Interdisciplinary Science Reviews, Vol. 18, No. 3, London, 1993
3. A. Müller, Parallelisierung numerischer Verfahren zur Beschreibung von Ausbreitungs- und chemischen Umwandlungsprozessen in der atmosphärischen Grenzschicht, PhD thesis, University of Karlsruhe, 1995

Parallel reduction of banded matrices to bidiagonal form

Bruno Lang

University of Wuppertal, Faculty of Mathematics

Project: Parallel Numerical Linear Algebra

Director: Andreas Frommer

Abstract. The project "Parallel Numerical Linear Algebra", lead by Andreas Frommer and Bruno Lang, is aimed at developing highly efficient parallel algorithms for solving eigenvalue, singular value problems and linear systems for dense and banded matrices [1,2]. In this note we shortly describe the reduction of banded matrices to bidiagonal form, which is an important preprocessing step in computing the singular value decomposition.

Problem description. To compute the singular value decomposition (*SVD*) $A = U\Sigma V^T$ of a matrix $A \in \mathbb{R}^{m\times n}$, one proceeds in two major steps. First, the matrix is reduced to bidiagonal form, $A = U_1 B V_1^T$, and then the Golub/Kahan procedure is used to compute the *SVD* $B = U_2\Sigma V_2^T$ of the bidiagonal matrix B. (Thus, $U = U_1U_2$ and $V = V_1V_2$.)

For banded matrices, the standard Householder reduction algorithm is not optimal because it does not make use of the banded structure. After a few steps, the sparsity of A is completely lost. Therefore, the number of operations and the memory requirements almost equal the costs for reducing a full matrix.

The "chasing algorithm" **_GBBRD** included in the LAPACK2.0 library preserves the banded structure of A by immediately removing *any* fill-in. In each step of this algorithm, a first rotation eliminates one element of the band (thereby generating one new off-band element), and then a sequence of rotations is used to chase the off-band element down along the band.

The reduction algorithm **HRed** developed in our project is based on Householder transformations. It also implements a chasing strategy, but it does not remove *all* fill-in. Thus, the memory requirements and the number of floating point operations for the reduction of A are somewhat higher than with **_GBBRD**. On the other hand, Householder transforms can be implemented using level 2 BLAS which usually perform significantly better than the level 1 rotations.

If the transformation matrices U and V are also needed then the overall number of operations in **HRed** is reduced by almost one third as compared to **_GBBRD**. In addition, the update of these matrices (involving the vast majority of the operations) can be done in a blocked fashion, therefore enabling the use of the level 3 BLAS.

Furthermore, **HRed** is well suited to parallel execution, as is demonstrated by numerical results in the next section. If the matrices are "reasonably large" then nearly full speedup can be obtained.

Numerical results. Figure 1 shows the overall performance attainable with different numbers p of processors. If the the matrix size is increased with p then the *performance per processor* is almost independent from the number of processors. Thus, the algorithm *scales* quite well.

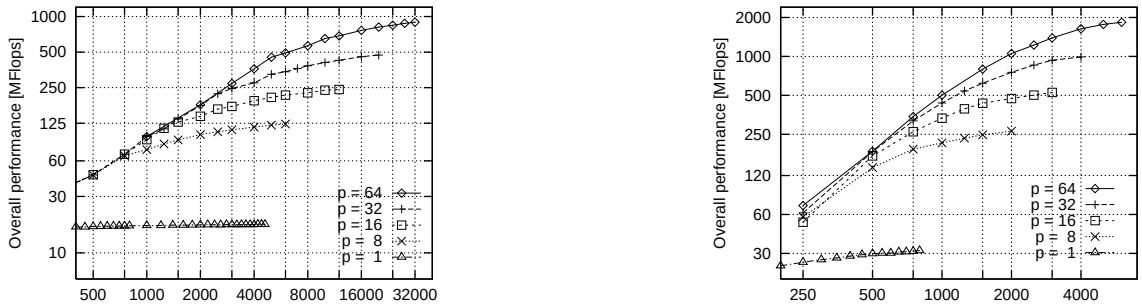


Figure 1: Overall performance of the parallel algorithm **HRed** for reduction alone (left diagram) and for reduction with update of *U* and *V* (right diagram)

References

1. B. Lang, A parallel algorithm for reducing symmetric banded matrices to tridiagonal form, SIAM J. Sci.Comput., **14**(6):1320-1338, November 1993
2. B. Lang, Parallel reduction of banded matrices to bidiagonal form, Parallel Computing, **22**:1-18, January 1996

Elastic sheet model of the microtubule wall

Imre M. Jánosi

High Performance Computing Centre (HLRZ), c/o Research Centre Jülich

Project: Computational physics at the HLRZ: Biologically motivated simulations

Director: Dietrich E. Wolf

The advent of electron microscopy and the development of sophisticated biochemical techniques has led to the realization that the cell sap contains a dynamically changing network of interconnected microtubules and filaments that together impart an architectural framework to the cell, the so-called *cytoskeleton*. The cytoskeleton serves two main purposes in cells. It is an internal scaffolding which supports and organizes a cell's interior, and gives a cell its characteristic shape. It also makes movement possible: Just like the skeleton of vertebrates is associated with muscles that implement movement of the body, the cytoskeleton contains motile elements that permit movement of the cell as a whole, as well as motion of intracellular components such as chromosomes, membranes, and granules. The first component of the cytoskeleton identified clearly by electron microscopy in the early 1960'es was the *microtubule* (MT), a long, rigid, rodlike structure. It is formed by "crystallization" of a protein, the *tubulin*, into a cylindrical lattice structure, usually consisting of 13 filaments, and having a diameter of 25 nm, while it can be up to several millimeters long. It is the most rigid element found in cells, and serves as a universal engineering element in nature's designs: Wherever a rigid rod, a pole, a beam, or a rail is needed in a cell, a microtubule is used. Recent high resolution micrographs taken at the European Molecular Biology Laboratory in Heidelberg show that the *end* of a growing MT is not cylindrical as the body of the tube is. Instead it consists of several long, outwards bent sheets, each sheet consisting of just a few connected protofilaments (Fig. 1). Many of these tips are geometrically shaped as sections of circles and display a constant mass density along their lengths. The simplest explanation of these observations is that the circular shape is inherent to these tips.

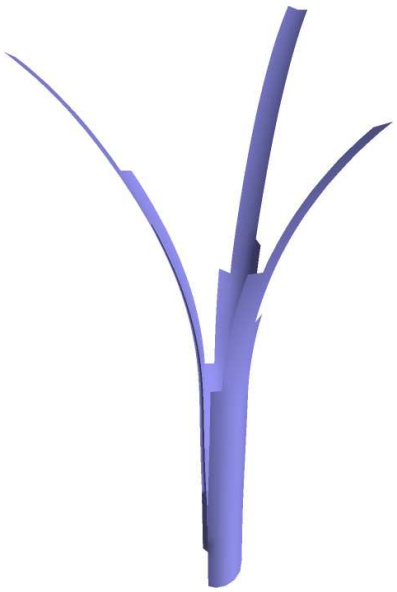


Figure 1: Elastic sheet model of a free microtubule end.

The outward bending should be the consequence of a competition in preferred bonding angles between the longitudinal and transverse tubulin bonds. We assume that the observed tip-shapes are manifestations of the inherent elastic properties of laterally bound protofilaments, and proceed to deduce these properties from the observed shapes. We assume that all filaments

have the same properties, and that any filaments have the same properties along its length.Then different circular-shaped tips can have different radii of curvature only because they contain different numbers of protofilaments. We represent the tubulin bond-network as a two-dimensional rectangular elastic sheet of the following properties. The tendency of spontaneous tube-forming is associated with a built-in curvature prescribed by a favored angle between neighboring chemical bonds in the transverse direction. The observations of exclusively outward bending MT-end structures suggest that a built-in curvature of opposite sign is present in the longitudinal direction too. The equilibrium shape of a free elastic sheet is determined by the minimum of the total energy, which is a sum of stretching and bending contributions. A general analytical solution might be based on a minimization of the total energy with respect to a large class of possible surface shapes. Although the theory of thin plates was established long ago, almost all of its huge literature is devoted to the special case of small deflections, which is clearly not adaptable in our case. To get a hint, what is the equilibrium shape of an elastic sheet with two competing built-in curvature, we performed direct lattice simulations.

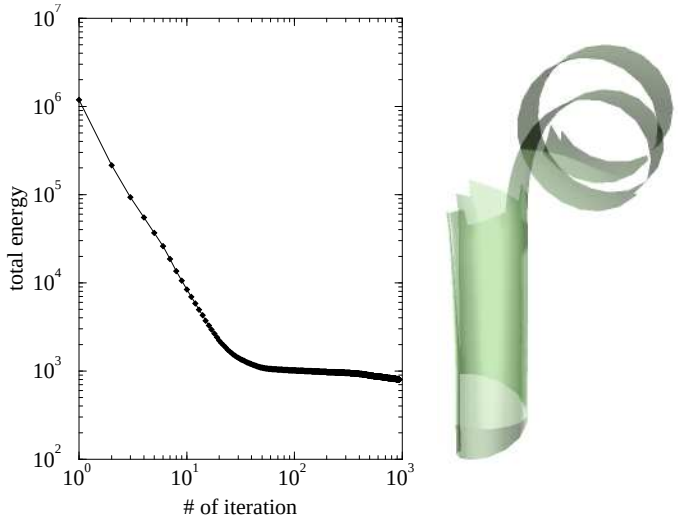


Figure 2: Total elastic energy relaxation for a twisted curly sheet.

Given a set of values for the bond lengths, intrinsic curvatures, and stiffnesses against their deformation, the preferred shape of a microtubule and its tip is found by finding the configuration of their triangulation which minimizes the associated energy. We do this with a conjugate-gradient relaxation method which involves all variables in each relaxation step; that is, updates are *global*. We found this algorithm to be the most efficient one after trying also several local algorithms. A typical amount of CPU time consumed in an optimization (e.g. Fig. 2) is 4-18 hours on a SUN Sparc 20 workstation, strongly depending on the lattice size. Considering the large parameter-space of the model, any systematic numerical analysis requires a large computational capacity. We used the Intel Paragon XP/S computer with 138 processors installed at the KFA/ZAM. The simplest possible way to parallelize the computation is also the optimal way: Each processor was left to do the entire energy minimization for one set of parameter values, with other processors doing the same calculation for different parameter values. Proceeding this way, each processor is approximately equivalent to one workstation.Based on our numerical analysis, we give a coherent explanation of the rich microtubule end morphology, and provide an order of magnitude estimation for the local elastic parameters.

References

1. D. Chrétien, S.D. Fuller, and E. Karsenti, *J. Cell Biol*. **129**, 1311 (1995)
2. Jánosi, D. Chrétien, H. Flyvjberg, and S. Leibler, in preparation

A model for microtubule oscillations

Elmar Jobs

High Performance Computing Centre (HLRZ), c/o Research Centre Jülich

Project: Computational physics at the HLRZ: Biologically motivated simulations

Director: Dietrich E. Wolf

In this project we are interested in numerical models of biological processes. One of the problems we work on comes from molecular biology of the cell and deals with the dynamic instability of microtubules. Microtubules are small, stiff tubes inside the cells. They are self-assembling polymers of a protein called *tubulin*. As generic engineering elements, they play an important role in the functioning of cells. Microtubules are the main part of the supporting structure of the cell and they act as beams and rails. But one of the most important functions of microtubules is during mitosis, the cell division cycle. Here, the microtubules start growing from the two centromers towards the chromosomes and attach to them. After all chromosomes are attached, the microtubules disassemble and pull the chromosomes apart. Now the cell can divide into two parts. All those mechanisms microtubules are involved in require that they are able to self-assemble and disassemble quickly and efficient. This ability is observed best *in vitro*. In vitro experiments use purified tubulin under fixed external conditions to examine the properties of these solutions. In a glycerol-buffered solution microtubules simply assemble by addition of free tubulin heterodimers in a stochastic process until the free tubulin is used up[1]. In an unbuffered solution, microtubules can exist in two states, a growing state as in the unbuffered system, and a shrinking state where the microtubules fall apart very rapidly. The microtubules change between those two states by stochastic transitions. This phenomenon is called *dynamic instability*. In vitro this effect can be observed very easily. Starting from a solution of tubulin with no microtubules present, they start to assemble and disassemble in phase. Because of the stochastic nature of the transition process the microtubule oscillations become dephased resulting in a damped oscillation of the total amount of polymerized tubulin. This can be observed for example in a light scattering experiment. An example curve is shown in fig.1.With a chemical reaction model based on the work of A. Marx and E. Mandelkow, we are able to explain the experimental curves quite well. Our model is based on an analysis of the experimental data with methods known from statistical physics. This analysis leads us to a formulation of the problem which includes the spatial distribution of the microtubules as well as the global concentrations of various products. The simple reaction model of Marx and Mandelkow now turns into a more detailed model consisting of a coupled set of nonlinear partial and ordinary differential equations. For a detailed description of this model see [2]. Although a minimal model, this problem is computationally extremely complex. The task to be solved is an optimization of the free parameters of the model. This is done by integrating the equations forward in time and comparing the result to the experimental data. The distance measure we used is χ^2. Being a function of the parameters of the theory, χ^2 has to be minimized for a given experimental curve. Although a minimal model, this problem is computationally extremely complex. The task to be solved is an optimization of the free parameters of the model. This is done by integrating the equations forward in time and comparing the result to the experimental data. The distance measure we used is χ^2. Being a function of the parameters of the theory, χ^2 has to be minimized for a given experimental curve. The model has 14 parameters of which only eight are approximately know from experiments. The other variables also have a biochemical interpretation, but they are not directly observable by the experiments.For those variables we have to find good starting values.

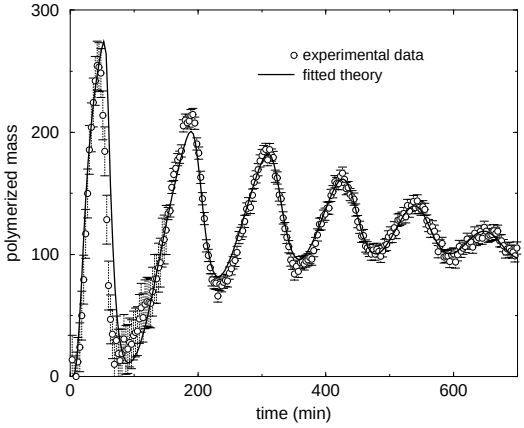


Figure 1: Dynamic instability of microtubule assembly. Experimental data from A. Marx and E. Mandelkow obtained by X-Ray scattering.The error-bars are rough estimates. For times smaller than 150 minutes we increased the error-bars to make the algorithm less sensitive in that region

A robust, but not very fast way to optimize a multidimensional problem with parameters which are not bound is the method of *Simulated Annealing*. This type of algorithms is easy to parallelize and performs very good on parallel computers. For a discussion of this topic see for example [3]. We took an approach based on this method but with a modified parallel implementation. A simulated annealing algorithm takes a step in random direction and random length in parameter space. The function to be minimized is evaluated at this point. If the result is smaller than the previous one, the step is accepted. Otherwise the step is accepted only with a certain (small) probability. This probability is slowly reduced within time. This gives the system the chance to escape from local minima for early times and fixes the solution at late times. On a parallel computer several trial steps can be evaluated at the same time and only the best solution is accepted. The smaller the acceptance probability, the higher the efficiency. For late times, when the algorithm is almost converged, the efficiency approaches 100%. The overall speedup strongly depends on the initial configuration, the annealing parameters, the number of nodes and the problem itself. We experienced total efficiencies between 60% and 90%. Each integration of the set of coupled ODEs and PDEs takes several minutes on a Sun SPARC 20 workstation. This order of magnitude was also to be expected on a single node of the Intel Paragon. We therefore implemented the algorithm in a master-slave model with software-interrupt handling. The master distributes the tasks (single sets of model parameters) to the slaves. As soon as a slave accepted a step, the result is sent to the master. The master then sends an interrupt to all slaves to stop their search. After confirmation of the interrupt, each slave receives a new task. The drawback of this algorithm is, that much more evaluations of the function (i.e. χ^2) are needed than for a simple steepest descent algorithm. Because the computational cost of a single integration is extremely large, we were not able to finish the optimizations within the given computer time. We therefore took intermediate results of the annealing which already restrict the parameter space we had to search and fed them to a steepest descent like algorithm on a vector-computer. We also used the Intel Paragon to gather the data for a visualization of the parameter space. It was done via a simple scan of the parameter space for a subset of parameters. Because the calculations of the model were completely independent of each other, the efficiency of this was 100%.

References

1. H. Flyvbjerg, E. Jobs and S. Leibler, Kinetics of Self-Assembling Microtubules: An "Inverse Problem" in Biochemistry, submitted to *Proc. Natl. Acad. Sci.* (USA)
2. E. Jobs,H. Flyvbjerg. An Extended Model for Microtubule Oscillations, to be published
3. E.H.L.Aarts, F.M.J.deBont, E.H.A.Habers, P.J.M. van Laarhoven. Parallel implementations of the Statistical Cooling Algorithm. *INTEGRATION, the VLSI journal* **4** (1986), 209

Quantum Monte Carlo simulations of random transverse Ising models

Heiko Rieger

High Performance Computing Centre (HLRZ), c/o Research Centre Jülich

Project: Quantum Griffiths Singularities

Director: H. Rieger

Strongly disordered systems show extreme sample to sample fluctuations at low temperatures. In particular many quantities of interest are not self-averaging, i.e. one large system does *not* provide the same information as many small samples. This means that in order to obtain reliable information not on typical but on average quantities one has to produce a good statistics via a large number of disorder realizations. As a consequence the simulation of thousands of medium sized samples is obviously better than trying to establish world records in system size for only one or two samples.

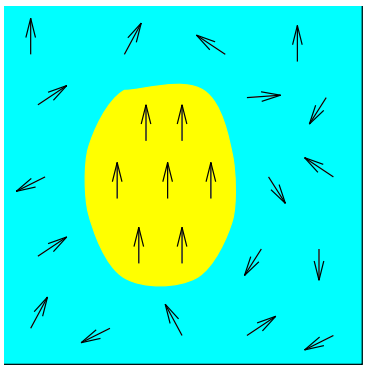


Figure 1: Sketch of a disordered magnetic system. The arrows inidicate magnetic moments. They interact with each other with varying ferromagnetic strength. In the yellow region these are strong, the tendency to ferromagnetic order (i.e. all arrows point in the same direction) is large, implying also large values for the local magnetic susceptibilities. In the green area this tendency is weak, so the arrows point in arbitrary directions a nd the local response to an external magnetic field is small.

Therefore the application is very simple but highly efficient: Since system size is not too large each sample fits into the memory of one processor, sample statistics is obtained by running the same program with different disorder realization on as many processors as one can get. Care has to be taken for the proper initialization of the random number generators on the different processors in order to generate statistically independent samples. But otherwise the machine runs with nearly the same code on a workstation and a massively parallel supercomputer, provided it makes the largest possible use of the capabilities of the corresponding CPUs. The algorithm we use is a straighforward single spin flip heat bath algorithm for Ising spins with tabulated exponential transition probabilities, periodic boundary conditions and sublattice update and uses exclusively integer arithmetic in the innermost loop. The performance of each processor for our code is comparable with that of a single Sparc 10 processor. Communication is done only at the beginning and the end of the simulation, by which the speed-up is maximal.

We investigated in this way the quantum phase transition and occurrence of Quantum Griffiths singularities in the two-dimensional Ising spin glass in a transverse field [1,2]. For this purpose we had to produce very accurate histograms for various local and global susceptibilities, which necessitates an ensemble size of up to 10.000 samples each one spending between 1 and 10 hours in a single CPU.

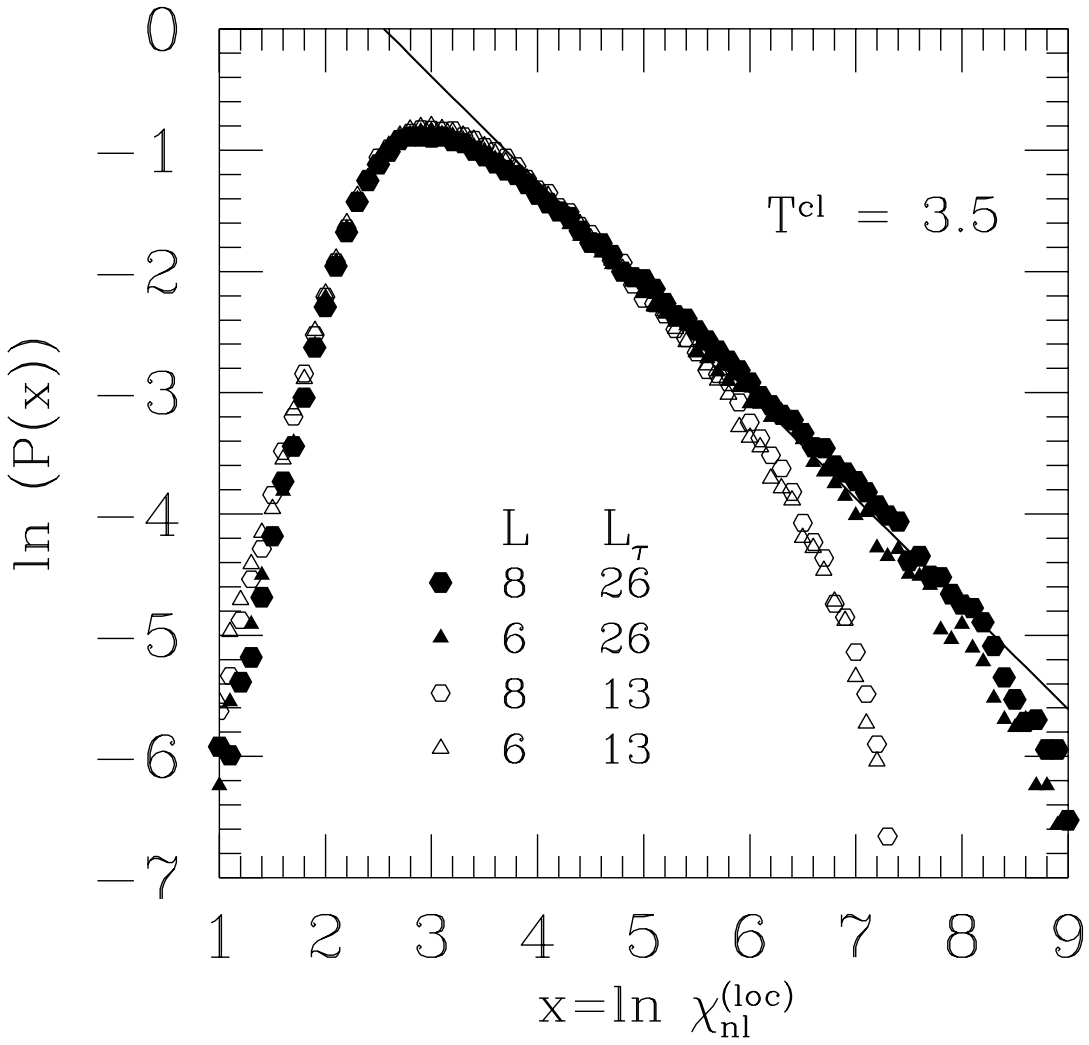


Figure 2: The log of the distribution of the logarithm of the local non-linear susceptibility of a two-dimensional quantum spin glass at very low temperatures and a noncritical value for the transverse field strength (i.e. within the so called Griffiths phase). Note the extremely wide range of the distribution (down to very small probabilities), which necessitates a huge number of samples (1000 to 10000), each needing roughly an hour of CPU time for equilibration. Increasing L_τ, i.e. decreasing temperature, seems to simply extend the range over which the data lies on a straight line. The solid line is a fit to the straight line region of the data and has slope that implies a divergence of the average suscptibility at zero temperature, which is the most remarkable result of this study.

References:

1. H.Rieger und A.P. Young, Zero temperature quantum phase transition of a two-dimensional Ising spin glass, Phys. Rev. Lett. 72, 4141 (1994)
2. H. Rieger and A.P.Young, Griffiths singularities in the disordered phase of a quantum Ising spin glass, Phys. Rev. B (1996), in press

Simulation of homoepitaxial growth

Harald Kallabis

High Performance Computing Centre (HLRZ), c/o Research Centre Jülich

Project: Computational physics at the HLRZ

Director: D. E. Wolf

In recent years there has been a great interest in understanding the growth of thin solid films. Such films are of broad technological importance as they form the base material for the construction of integrated circuits, sensors or solid state lasers to name only a few.

By now, there are several experimental techniques for deposition of atoms on a substrate from the gas phase, examples of which are chemical vapour deposition (*CVD*), sputter deposition and molecular beam epitaxy (*MBE*). Those differ in the quality of the deposited films in terms of defects, surface smoothness and incorporation of foreign atoms on one hand and costs on the other.

However, when highly controllable growth is the main objective, MBE is the best choice by far. Thus, understanding MBE growth is of outstanding importance in the field of research on the structure of thin solid films for the different applications mentioned above.

The simplest case of epitaxial growth is when the substrate and the deposited atoms are of the same material (*homoepitaxy*). Here, the important processes are deposition of atoms on the substrate, which is a process random in time and space, and diffusion of adsorbed atoms which is thermally activated and therefore also random.

Then, one can distinguish three different stages of growth: the submonolayer regime, where one finds a typical distance between the islands grown in the first layer, the layer-by-layer growth regime, where physical observables like for example the surface width or the Bragg intensity show oscillatory behaviour over several deposited layers and finally the kinetic roughening regime with a characteristic power law dependence of the observables on time and space [1] (Fig. 1).

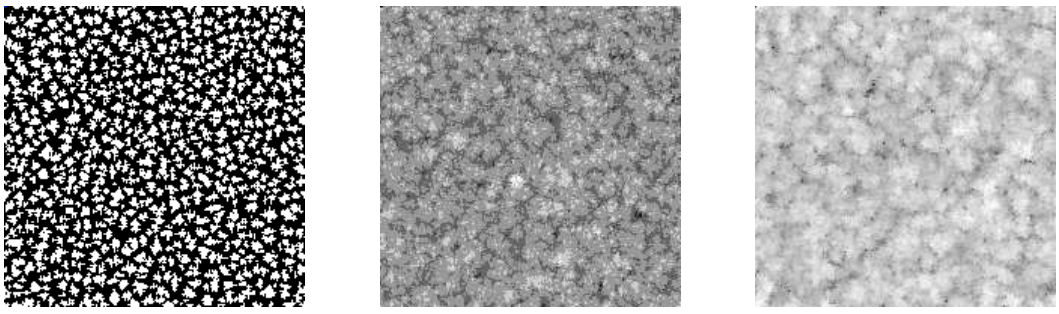


Figure 1: Snapshots of a growing surface in different stages of growth (from left to right): submonolayer (0.5 monolayers (MLs) deposited), layer-by-layer (30 ML) and kinetic roughening (2000 ML) regime. The lowest exposed layer is black, the highest exposed layer is white.

The desired growth mode for producing thin films on perfectly oriented substrates (i.e. without a miscut) is layer-by-layer growth, so that one has to understand for how many monolayers this growth mode prevails before the kinetic roughening regime takes over. This number of

monolayers N_c is expected to be a function of the ratio of the diffusion constant D of the adsorbed atoms on the substrate to the deposition rate F. A detailed analysis of the equations believed to describe growth in the kinetic roughening regime predicts a power law dependence with an exponent $\frac{1}{3}$ for a one dimensional substrate and $\frac{2}{3}$ for a two dimensional substrate [2]. Computer simulations of the one dimensional case performed on the Paragon and on Sun Sparc 20 workstations support this analysis (Fig. 2).

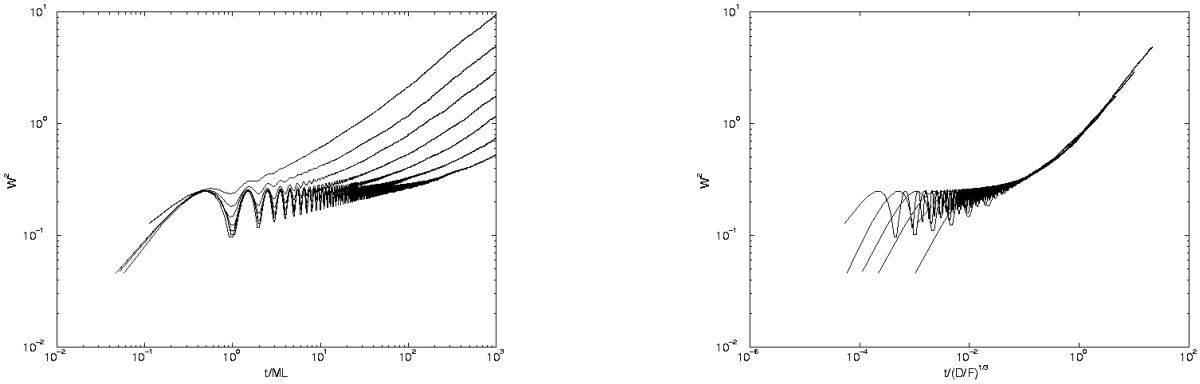


Figure 2: Computer simulation results of epitaxial growth on a one dimensional substrate for the surface width as a function of time for various parameter values D/F: unscaled time (left) and time scaled with $D/F^{1/3}$ (right).

The computer simulations generically involve averaging of observables over many realizations of deposition events (being random in time and space) and random hops of adsorbed atoms. These different realizations have to be independent and therefore this problem is ideally suited to be run on a parallel computer, where the run time is reduced by the number of available nodes. For this type of problem the performance of one node on the Paragon can be compared with that of a Sun Sparc 20 workstation, so that the simulations are finished typically 138 times earlier on the Paragon with 138 nodes than on a single workstation. Run times for one of the simulations (Fig. 2) are 5 to 500 CPU hours, depending on the parameter D/F.

References:

1. D. E. Wolf, in *Scale Invariance, Interfaces, and Non-Equilibrium Dynamics*, eds. A. McKane et al., Plenum Press, New York, 1995
2. L. Brendel, H. Kallabis, J. Krug, M. Schroeder, P. Smilauer, D.E. Wolf, J. Villain, in preparation

Susceptibilities and cumulants in two-flavour QCD

Frithjof Karsch, Edwin Laermann and Manfred Oevers

University of Bielefeld, Faculty of Physics

Project: Quantum chromo dynamics

Director: F. Karsch

At high temperature, Quantum Chromodynamics (QCD), the theory of the strong interactions between quarks, predicts a phase transition from "normal" hadronic matter to a quark gluon plasma phase. This phase transition has very likely taken place during the cooling of the early universe. Heavy ion collision experiments at the accelerators RHIC and LHC aim at reproducing this phenomenon in the laboratory. For an understanding of the transition and of the experimental data it is important to know in detail the dynamics of quarks and gluons at and in the vicinity of the critical temperature, i.e. the critical behavior of the theory.

An important feature of the transition is the restoration of chiral symmetry in the high temperature phase. In contrast to QCD with three or more light flavour degrees of freedom, where numerical simulations seem to indicate that the chiral symmetry is restored abruptly (first order transition), there is increasing evidence that the two-flavour theory has a second order phase transition in the limit of vanishing quark masses. In fact, recently it was shown that the functional behaviour of various observables is consistent with the scaling behaviour for a chiral phase transition with critical exponents in the universality class of a 3-d *O(4)* σ-model.

This behaviour is derived from the scaling of the singular part of the free energy

$$f(t,h) \equiv -\frac{T}{V} \ln Z = b^{-1} f(b^{y_t} t, b^{y_h} h).$$

where t is the reduced temperature, $t=(T-T_c)/T_c$, h is proportional to the quark mass, $h = m_q/T$, b is an arbitrary scale factor and y_t, y_h denote the critical exponents. For example, the chiral susceptibility χ_m, is obtained as the second derivative of the partition function Z with respect of the quark mass,

$$\chi_m = \frac{T}{V} \frac{\partial^2}{\partial m_q^2} \ln Z$$

At the transition, the susceptibility develops a peak the location of which allows one to give a precise quantitative definition of the (pseudo) critical temperature at non-vanishing values of the quark mass. Moreover, the peak height scales according to

$$\chi_{m,max} = c_m \, m_q^{1/y_h - 2}$$

This and similar relations provide information on the critical exponents as well as on the scaling of the (pseudo) critical temperature with quark mass.

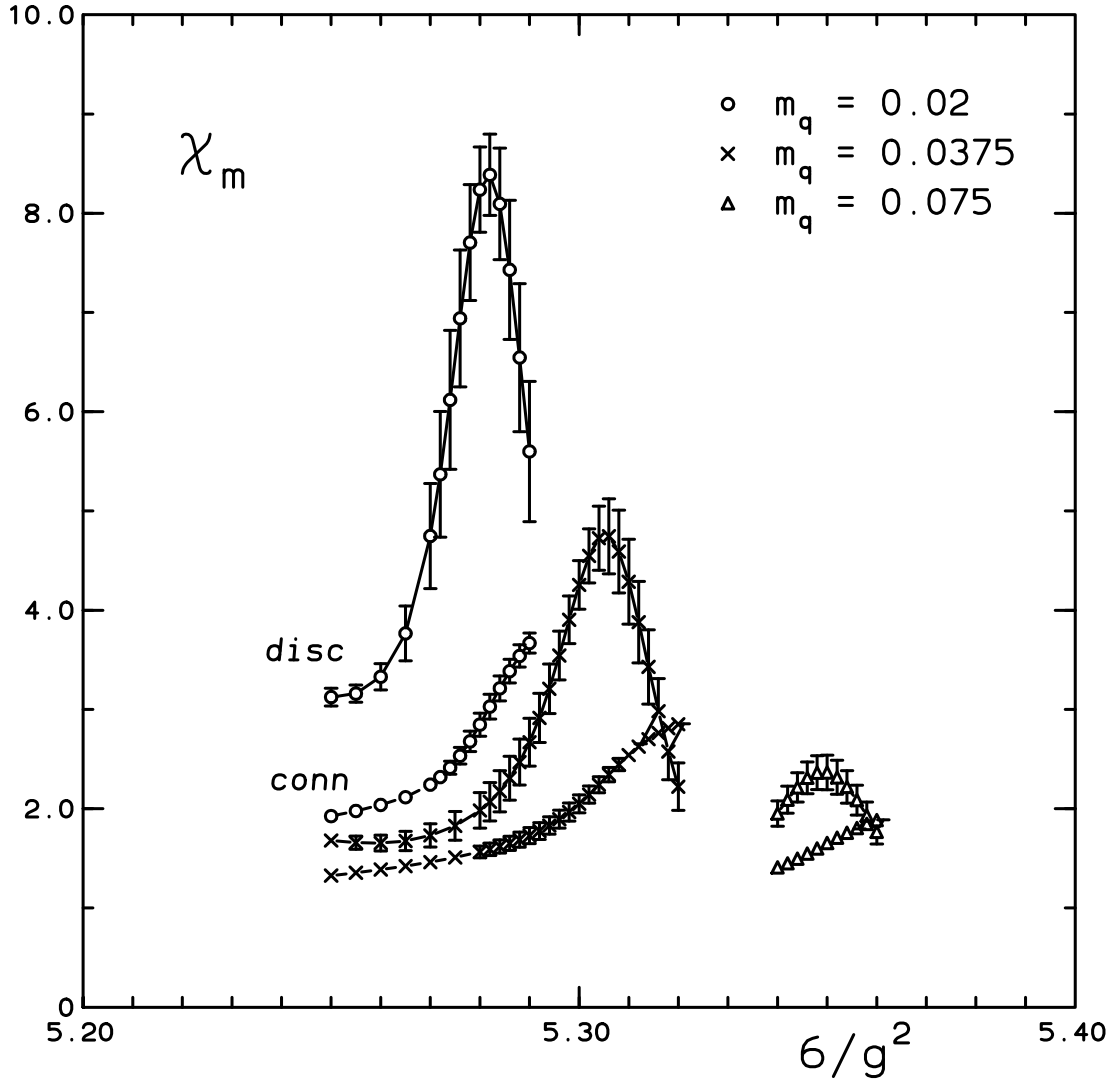


Figure 1: The chiral susceptibility

This has been exploited so far on small lattices ($8^3 \times 4$) only. One of the results, the chiral susceptibility is shown in the figure.In the PARAGON project we analyze lattices of larger size in order to check for (the absence of) finite size effects.

References:

1. F. Karsch, Phys. Rev. D49 (1994) 3791
2. F. Karsch and E. Laermann, Phys. Rev D50 (1994) 6954
3. R.D. Pisarski and F. Wilczek, Phys. Rev. D29 (1984) 338
4. F. Wilczek, Intern. J. Mod. Phys. A7 (1992) 3911
5. K. Rajagopal and F. Wilczek, Nucl. Phys. B399 (1993) 395

PARCOMB: A parallel code for the simulation of reactive flows

Marc Lange, Uwe Riedel, Dominique Thévenin, Jürgen Warnatz

University of Heidelberg, Interdisciplinary Center for Scientific Computing (IWR)

Project: Simulation of flame instabilities

Director: J. Warnatz

The numerical simulation of turbulent reacting flows remains up to now one of the most challenging tasks in the field of high-performance computing. In addition to the classical four variables (density, *x*- and *y*-components of velocity, total energy) needed for two-dimensional computations of laminar non-reacting flows hundreds of reactive species have to be computed and stored at each time-step. In typical cases, characteristic time-scales associated with this kind of problem may vary between 10^{-8} ~s for some important intermediate radicals and 1 s for some slowly-produced pollutants like NO. It is the same for characteristic lengths, ranging from the Kolmogorov length (i.e. the smallest possible length scale associated with turbulence features – typically a few micrometers –) to the total length of the computational domain, which should be at least of the order of a centimeter. Due to these requirements an accurate simulation of turbulent reacting flows can only be achieved at high computational costs for the time integration (very stiff equations) and for the spatial resolution (very different length-scales). This leads to the conclusion that such computations cannot be carried out on classical vector supercomputers in a reasonable time.

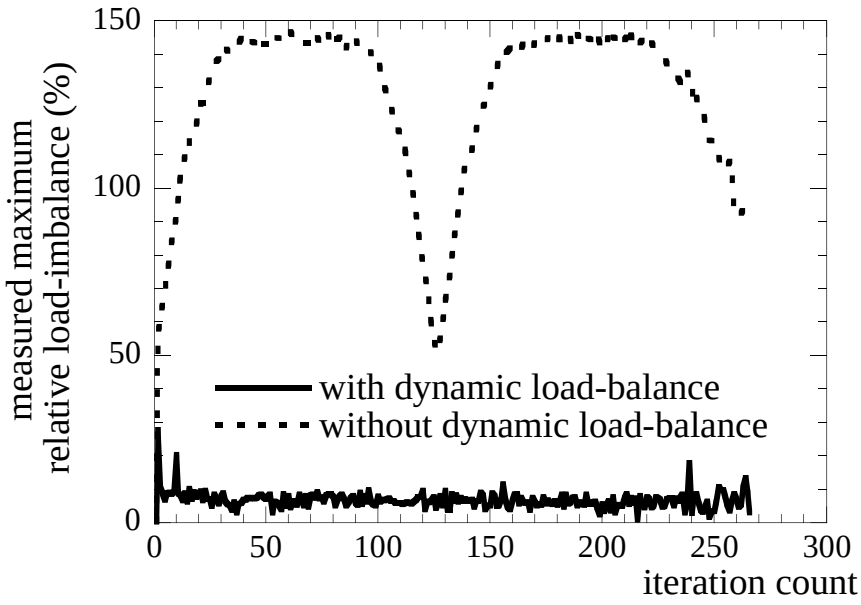


Figure 1: Observed relative load-imbalance between processors with and without dynamic load-balancing

We have developed a computer code for Direct Numerical Simulation (DNS) on massively parallel computers. This code uses detailed transport, thermodynamic and reaction models. It works with a fully explicit two-dimensional finite-difference solver with sixth-order spatial resolution, which is needed to be able to resolve the smallest structures of the flow. Integration in time is carried out using a fourth-order Runge-Kutta method. The parallelisation is based on domain decomposition methods. To get a better load balance the grid-points are dynamically

redistributed between the processors during the computation. As shown in figure1 this procedure leads to a much more equal utilization of the processors and as a result to a higher efficiency.

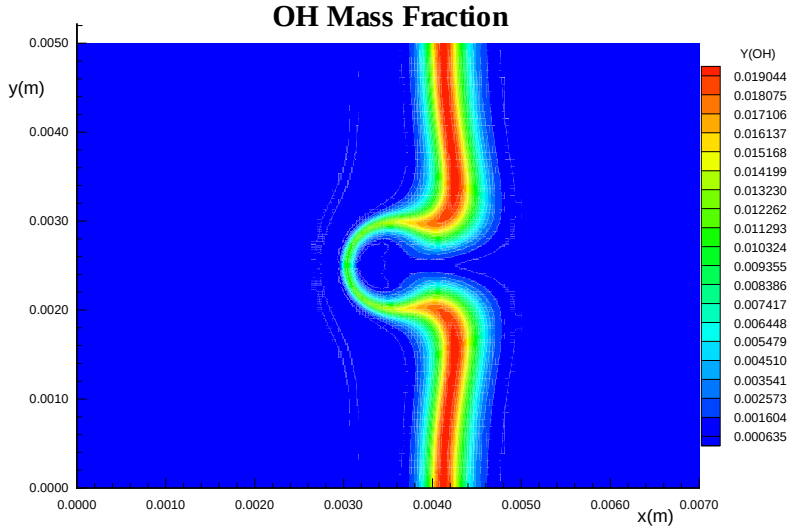


Figure 2: OH mass fraction in a hydrogen-air diffusion flame interacting with a vortex-pair

We apply this DNS code to investigate the interaction between combustion-processes and turbulent flow-fields. Detailed knowledge of these interactions is necessary for the understanding of pollutant formation. As an example application figure 2 shows the distribution of the OH mass fraction in the case of a vortex-pair crossing a hydrogen-air diffusion flame. This simple configuration corresponds to a basic feature of turbulent reacting flows as turbulence may be modelled as a collection of differently sized vortices.

References:

1. Warnatz J., Maas U., Dibble R.W.:Combustion, Springer-Verlag (1996)
2. Thévenin, D., Behrendt, F., Maas, U. and Warnatz, J.: Dynamic load balancing for parallel simulation of reacting flows, Proceedings Parallel CFD '94, (A. Acer, P. Periaux and N. Satofuka, Eds.), Elsevier, (1995)
3. Thévenin, D., Behrendt, F., Maas, U. and Warnatz, J.: Simulation of reacting flows with a portable parallel code using dynamic load-balancing, Proceedings High-Performance Computing and Networking Europe '95, Lecture Notes in Computer Science **919**, (Hertberger, B. and Serrazi, G., Eds.), Springer, pp.378-383 (1995)

Current density distributions in semiconductor devices and gas discharge systems

M. Or-Guil, Ch. Schenk, F.-J. Niedernostheide, H.-G. Purwins

University of Münster, Institute of Applied Physics

Project: Semiconductors

Director: H.-G. Purwins

We investigate a set of two integro-differential equations that show several analogies to reaction-diffusion equations of activator-inhibitor-type and that are useful to describe self-organisation of charge carrier density, current density and potential distributions in p-n-p-n diodes and special gas discharge systems [1]. The quasi-particle behaviour of localized current density filaments in a spatially two-dimensional region is numerically analysed and compared with experimental results. Systematic parameter variations and a detailed spatio-temporally resolved analysis of the effects of colliding localized current filaments are carried out to clarify the basic physical mechanisms of specific effects in both systems [2]. A further point of main effort concerns numerical simulations in the vicinity of bifurcation points in order to determine scaling-laws to allow quantitative comparison with experiments.

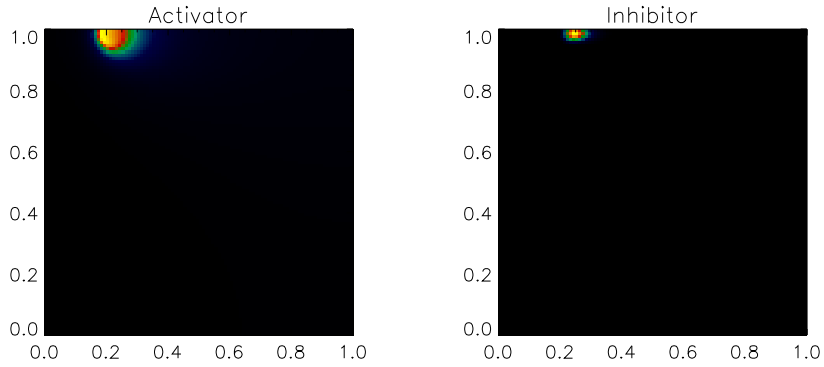


Figure1: The numerically calculated distribution of the activator, i.e. the potential, and the inhibitor, i.e. the current density, of a moving filament in a quadratic region is shown. The filament will collide with a virtual, mirror-image filament on the edge, being reflected afterwards. White represents the maximum, blue the minimum potential and current density value respectively. The calulation models the behaviour of the current density in a p-n-p-n diode.

Figure 1 shows a filament, consisting of a localized high potential (activator) and current density (inhibitor) distribution moving towards the edge of a quadratic region. Since the system has no-flux boundary conditions, which act as a mirror, we are able to interpret the above situation as a collision of two filaments and so to investigate the interaction between filaments and its dependence upon varying parameter sets [3].

The numerical calculations consist of computing the model partial differential equations on a Paragon XP/S using a finite difference method. Time discretization is realized through the Crank-Nicholson method, the resulting implicit equations being solved by fixed point iteration. This algorithm is very suitable for parallelisation: The discretized, two-dimensional, rectangular space can be diveded in suitable subregions, each subregion being assigned to a node for computation. The spatial dynamics of the system is defined by a local as well as by a global, integral spatial coupling. This means that the boundary of the subregions are communicated to

the nodes which compute neighbouring subregions after each step of the fixed point iteration while global coupling information is communicated to all nodes after each time-step. As the boundaries of the subregios are calculated first and send-commands to neighbouring nodes are performed before the calculation of the inner part of the subregion, advantage can be taken of asynchronous communication commands, as shown in figure 2, and a speedup of e.g. 61 for 64 nodes in a 400 x 400 grid region can be obtained.

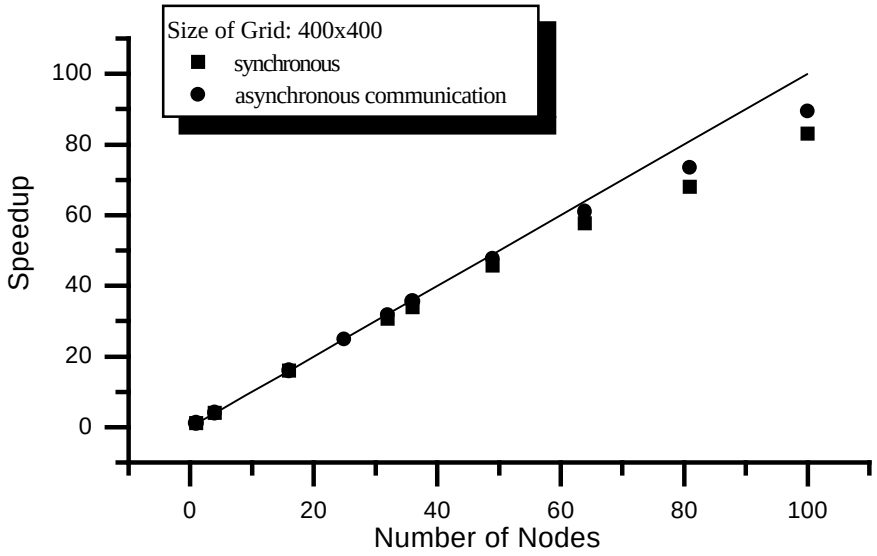


Figure 2: The graph shows the speedup of an exemplary calulation of current density filamentation in a p-n-p-n diode for different numbers of nodes but a constant grid size of 400 x 400. Results for both synchronous and asynchronous communication commands are shown. The solid line represents ideal parallelisation.

References

1. F.-J. Niedernostheide, B. S. Kerner, H.-G. Purwins, "Spontaneous Appearance of Rocking Localized Current Filaments in Nonequilibrium Distributive Systems", Phys. Rev. B 46, 7559 - 7570 (1992)
2. F.-J. Niedernostheide, M. Ardes, M. Or-Guil, H.-G. Purwins, "Spatiotemporal behavior of localized current filaments in p-n-p-n diodes: Numerical calculations and comparison with experimental results", Phys. Rev. B 49, 7370 - 7384 (1994)
3. F.-J. Niedernostheide, M. Or-Guil, M. Kleinkes, "Investigations on Two-Dimensional Patterns in a Multilayered Semiconductor Device" in: "Selforganization in Activator-Inhibitor Systems: Semiconductors, Gasdischarge and Chemical Media", Edts: H. Engel, F.-J. Niedernostheide, H.-G. Purwins, E. Schvll; Wissenschaft und Technik, Berlin (1996); in print

Gossiping, min-cost flow and list ranking on interconnection networks

Jop F. Sibeyn

Max-Planck-Institute for Computer Science, Saarbrücken

Project: Algorithms and complexity

Director: K. Mehlhorn

Introduction. We are using the Paragon facility at Jülich since half a year. We should underline that we do not use a parallel computer because we have some large problem to be solved, but because we investigate on parallel algorithms. Our experience on the Paragon gives us feed-back how to improve these. We have considered a communication problem called 'gossiping', a parallel algorithm for min-cost flow, and several algorithms for list ranking. We give an outline of our results in these directions.

Gossiping. Gossiping is one of the fundamental communication problems. It appears in many contexts, both theoretical and practical. Gossiping is the problem in which every PU of a network wants to send a packet of a given size to every other PU. Said differently, initially each of the N PUs of a network holds an amount of data of size L, and finally all PUs must know the complete data of size $N \times L$. This is a very communication intensive operation. For meshes with worm-hole routing it is not obvious how to organize the routing such that the total cost is minimal.

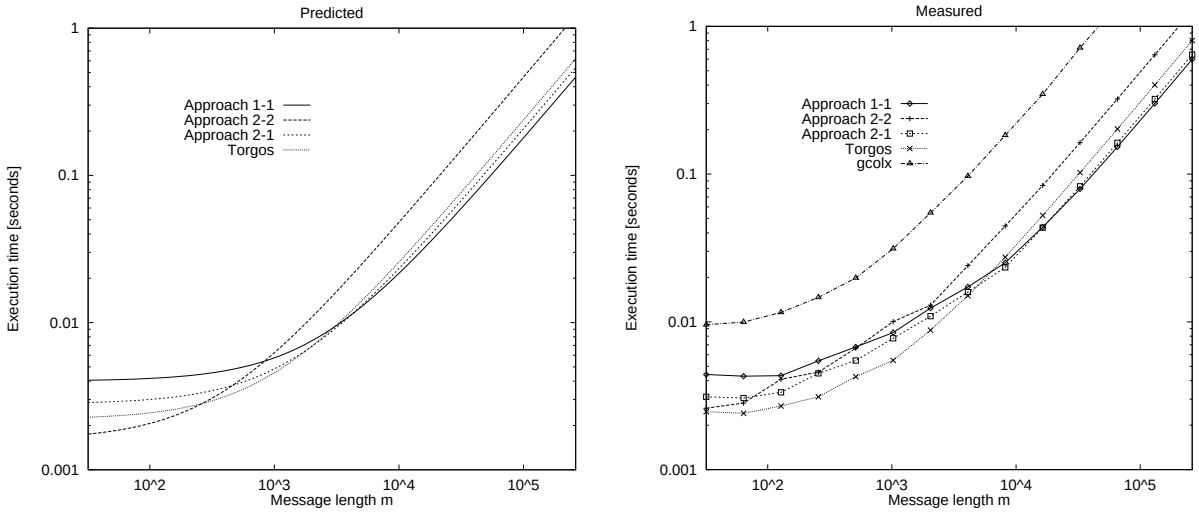


Figure 1: Predicted and measured execution times of the four one-dimensional gossiping algorithms on a 9×9 mesh

Together with Ben Juurlink (High Performance Computing Division, Dept. of Computer Science, Leiden University), who did the actual programming, I have implemented eight algorithms: four one-dimensional (1-D) gossiping algorithms and four two-dimensional (2-D) variations. These experiments were performed on 9×9 partitions of the Paragon supercomputer in Jülich. The most remarkable conclusion is that even our most basic algorithm was considerably faster than the standard gossiping routine of the system. This becomes clear from figure 1, where we also compare the performance of our implementations with the performance the global communication routine !gcolx!. For example, for messages of 32 KB, the !gcolx! routine requires 716 milliseconds, while Approach 1-1 requires only 79 milliseconds. Details are given in [2].

Min-Cost Flow. Network-flow problems have wide applications in several fields of great economical importance. Min-cost flow, is the problem in which a flow, of a certain prescribed total volume, has to be transferred through a flow network in which the channels do not only have limits on their capacities, but have also non-uniform costs.

This is a well-known problem, and a considerable amount of research has been performed before. Together with a student, who is preparing a masters thesis on this topic, we consider a fairly original algorithm for finding a near-to-optimal solution. The results are still preliminary, but rather promising. In any case, it already appeared that the amount of parallelization that can effectively be used. The optimal number of processors lies close to the expected degree of the nodes in the network.

List Ranking. A list is a basic data structure: it consists of nodes which are linearly linked together. The list ranking problem consists of determining the rank for all nodes. The rank of a node is its distance to either the first or last node of its list (these notions can be easily converted into each other). List-ranking is a very non-local problem, that resists efficient external-memory and parallel solutions. At the same time, it is a fundamental problem with wide applications. It can be considered as a measure for the power of middle-grain parallelism as we propose it.

Currently Pierre Kelsen, two masters students, and myself, are implementing and refining four fundamentally different algorithms. A 'one-dimensional' approach is ready and under testing: a list of 10^7 nodes can be ranked in about 15 seconds. A simulation makes clear for which choices of the parameters each of the algorithms performs the best. Results are given in table 1.

	The case r = 250											
512	p	p	p	p	p	i	i	s	s	s	s	s
256	p	p	p	p	p	p	s	s	s	s	s	s
128	p	p	p	p	p	p	s	s	s	s	s	s
64	p	p	p	p	p	p	p	s	s	s	s	s
32	p	p	p	p	p	p	p	p	s	s	s	s
16	p	p	p	p	p	p	p	p	s	s	s	s
8	p	p	p	p	p	p	p	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1
	1	**2**	**4**	**8**	**16**	**32**	**64**	**128**	**256**	**512**	**1024**	**2048**

Table 1: Vertically the sidelength of the two-dimensional mesh is given, horizontally the number of nodes per processor. ‘p’, ‘l’, ‘i’ or ‘s’ indicate that pointer jumping, the linear approach, independent-set removal or sparse-ruling set approach, respectively, are the best.

References

1. Sibeyn, J.F., List Ranking on Interconnection Networks, Techn. Rep. 11/1995, SFB 124-D6, Universität Saarbrücken, Saarbrücken, Germany, 1995
2. Juurlink, B., P.S. Rao, J.F. Sibeyn, Gossiping on Meshes and Tori, submitted to EUROPAR’96.
3. Kelsen, P., J.F. Sibeyn, List Ranking as Fast as Routing, submitted to SWAT ‘96

First-principles tight-binding electronic structure calculations

R. Zeller

Research Centre Jülich, Institute for Solid State Research

Project: Electronic structures of solids, surfaces and layers

Director: H. Müller-Krumbhaar

In the last decade, density-functional electronic-structure calculations have very much improved the ability to predict and explain materials properties from quantum-mechanical first principles. For larger systems, however, with more than a few tens of nonequivalent atoms, the solution of the density-functional equations requires enormous computer resources. Therefore, the electronic structure for large systems is presently still determined almost exclusively by empirical tight-binding methods, where interactions are assumed to involve only nearby atoms.

The tight-binding picture is also very often applied to understand and explain the quantum-mechanical behavior of electrons in atoms, molecules, and solid materials. Last year, an exact transformation of the density-functional equations into a tight-binding form with exponentially decaying tight-binding parameters was discovered [1] and it can be expected that this transformation represent a major breakthrough in tight-binding electronic-structure calculations. The applications on the Intel Paragon XP/S investigated how accurately this localization works in actual numerical calculations and how well the discovered method performs for large systems. In calculations of electronic densities of states as function of energy for copper and for free space (which is the most demanding system for any tight-binding description) the method was found to give accurate results for occupied and unoccupied electronic states at least up to 20 eV above the vacuum level. The density-of-states calculations are independent for each energy and thus completely parallelized. The applicability to large systems was studied in total-energy calculations for copper in supercell geometry. Figure 1 shows that the results agree with ones of a conventional band-structure calculation within an error of about 0.00003 eV per atom. (The absolute total energy for copper is 44571 eV per atom). The study also showed that convergence in supercell calculations is governed by the product of the number N of atoms and the number *k* of wavevectors used in the necessary integrations over the unit cell of the reciprocal lattice.

The computing time for large systems is essentially spent in inverting large matrices of dimensions 16 times *N*. Here the ScaLAPACK library routines [2] can be used with success as shown in figure 2 where speedup results are displayed assuming that the computing times scale with *N* cube. In future calculations considerable savings of computing time can be expected because the exponential decay leads to sparse matrices. Except for surface and interface calculations where the effort scales linearly with system size [1], it is, however, not yet clear how the sparseness can be exploited most efficiently.

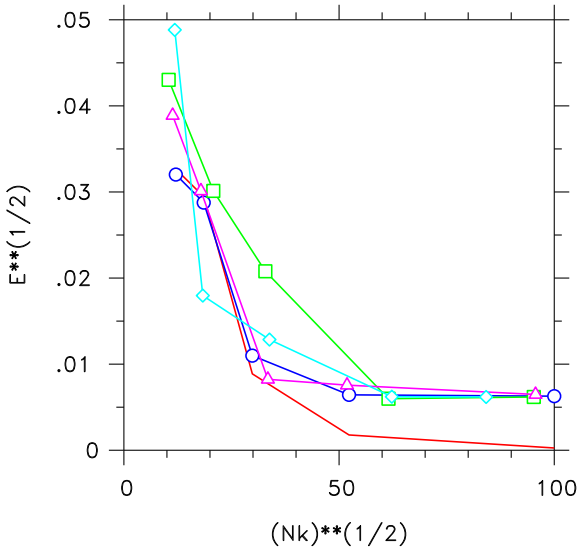


Figure1: Square root of the absolute total-energy error E per atom in units of $(eV)^{1/2}$ as function of the square root of the product of the number *k* of wavevectors and the number *N* of atoms per supercell. The lines with circles, diamonds, triangles, and squares are for *N*=1, *N*=4, *N*=32, and *N*=108. The line without symbols is for a conventional band-structure calculation.

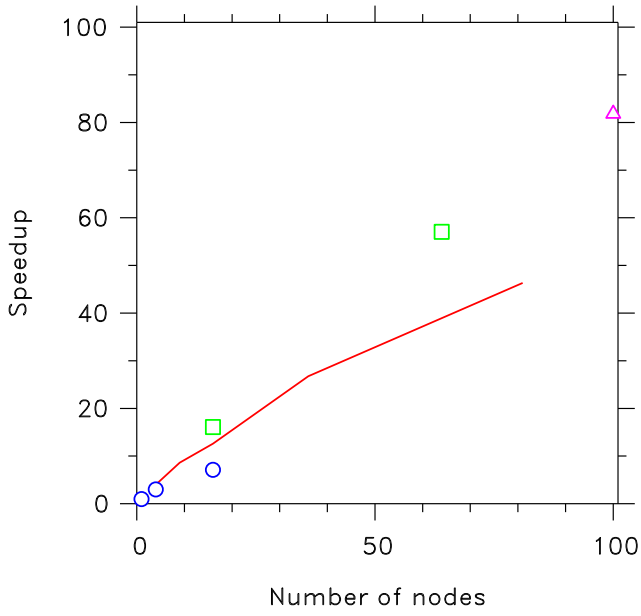


Figure 2: Speedup as function of the number of nodes for 108 atoms per supercell (solid line). Circles, squares, and triangle are for 32, 256, and 500 atoms assuming that the CPU times scale with the cube of the number of atoms.

References:

1. R. Zeller, P.H. Dederichs, B. Ujfalussy, L. Szunyogh, and P. Weinberger, Theory and Convergence Properties of the Screened Korringa-Kohn-Rostoker Method, Phys. Rev. B 52, 8807 (1995)
2. J.Choi, J. Demmel, I. Dhillon, J. Dongarra, S. Ostrouchov, A. Petitet, K. Stanley, D. Walker, and R.C. Whaley. ScaLAPACK: A Portable Linear Algebra Library for Distributed Memory Computers - Design Issues and Performance. Technical Report UT CS-95-283, LAPACK Working Note No. 95, University of Tennesee (1995)

Phase Field Models for Pattern Formation with Coupled Diffusion Fields

Th. Abel and H. Müller-Krumbhaar

Research Centre Jülich, Institute for Solid State Research

Project: Pattern Formation

Director: H. Müller-Krumbhaar

A thermodynamically consistent phase field model for description of pattern formation in a two component system with material and heat transport is deduced.The phase field Φ and the concentration field C and the temperature field T couple only at the interface. Outside the interface the phase field is constant and the behavior of the C and T field is given by diffusion equations. (The phase field Φ is used to distinguish the two states 'SOLID' (Φ = -1) and 'LIQUID' (Φ = 1).)

A binary mixture with parallel solidus and liquidus line (miscibility gap = 1) can be described by a free energy density

$$F = \frac{\xi^2}{2}(\nabla\Phi)^2 + V(\Phi) + M_0\left(\Gamma_L\left(\frac{1}{2}(C-2)^2 + \frac{1}{2}(Q-1)^2\right) + (1-\Gamma_L)\left(\frac{1}{2}(C-1)^2 + \frac{1}{2}Q^2 + 1\right)\right)$$

$$V(\Phi) \equiv V_0\left(\Phi^2-1\right)^2 \quad ; \quad \Gamma_L \equiv \frac{1}{2} + \frac{\Phi}{1+\Phi^2}.$$

Neglecting cross terms the dynamics of the conserved quantities C and Q ($Q \equiv T+\Gamma_L$: energy density) and the nonconserved quantity Φ is given by

$$\frac{\partial C}{\partial t} = \nabla D_C \nabla \frac{\delta \mathcal{F}}{\delta C} \quad ; \quad \frac{\partial Q}{\partial t} = \nabla D_Q \nabla \frac{\delta \mathcal{F}}{\delta Q} \quad ; \quad \frac{\partial \Phi}{\partial t} = -\tau \frac{\delta \mathcal{F}}{\delta \Phi}$$

i.e. the dynamics of the system can be described by a set of three coupled differential equations:

$$\begin{aligned}
\tau\frac{\partial \Phi}{\partial t} &= \xi^2\nabla^2\Phi + V_0(\Phi - \Phi^3) + M_0\,\frac{\partial \Gamma_L}{\partial \Phi}\,(C + T + \Gamma_L(\Phi) - 1) \\
\frac{\partial C}{\partial t} &= \nabla\left(D_C \nabla\left(C - \Gamma_L\right)\right) \\
\frac{\partial T}{\partial t} &= \nabla\left(D_T \nabla T\right) + \frac{\partial \Gamma_L}{\partial t}
\end{aligned}$$

Figure 1 shows contour plots of the three different fields at the end of a simulation. At the beginning a small solid circle with a threefold perturbation was put in the middle of the mesh.

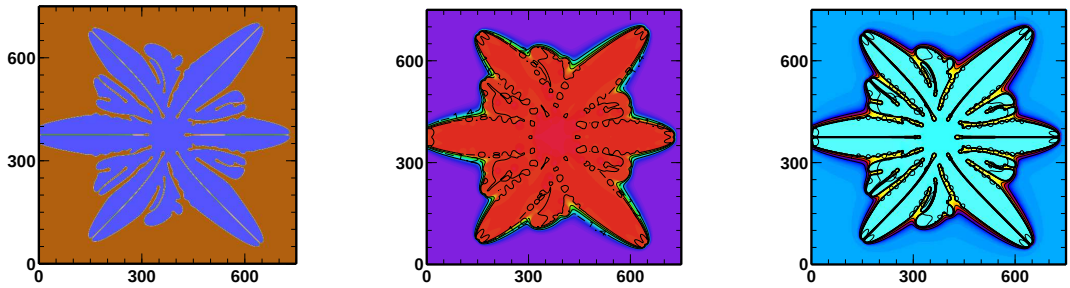


Figure 1: Phase field, temperature field, and concentration field of the final stucture; at the beginning of the simulation a small perturbated circle was initialized in the middle of the mesh.

The Alternating Direction Implicit Method is a suitable method to integrate this system on a single processor computer, but for numerical integration on the Intel Paragon parallel computer the explicit integration is preferable: the integration mesh can be divided into subgrids, every node solves the equations for one of these subgrids, and message passing is only necessary to update the boundary values of every subgrid. The algorithm was improved considering the fact that the right hand side of the phase field equation vanishes far away from the interface. Figure 2 shows the decrease of the real computing time with the number of used nodes. A calculation with 128 nodes is approximately 78 times faster than a calculation with one node.

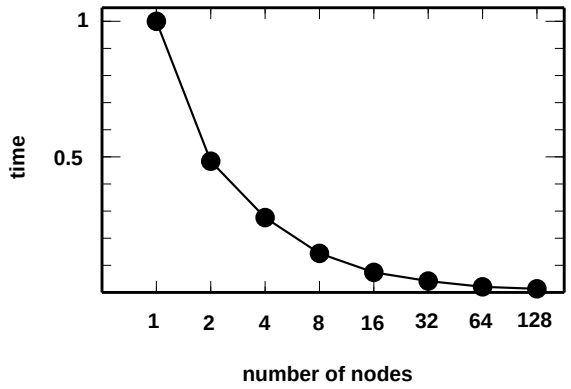


Figure 2: Computing time (related to the computing time with one node) for the integration in dependence on the number of used nodes; mesh size: 512x600; number of iteration steps: 1000

References:

1. Th. Abel, H.Müller-Krumbhaar, to be publised
2. A.Boesch, H.Müller-Krumbhaar, O.Shochet, Z.Physik **B 97** 367 (1995)
3. Th. Ihle, H.Müller-Krumbhaar, Phys. Rev. **E 49**, 2972 (1994)

Parallelization and load balancing of a comprehensive atmospheric chemistry transport model

Hendrik Elbern

University of Cologne, Institute for Geophysics and Meteorology

Project: EURAD

Director: A. Ebel

Numerical modelling of anthropogenic emissions into the atmosphere, their chemical conversions, long-range transport, and subsequent deposition to the surface has been recognized as an excellent tool to both analyze actual air pollution and simulate efects of abatement strategies. Focal points in this research area are acid precipitation and episodes of highly elevated ozone concentrations. In the present application an atmospheric chemistry-transport model (CTM) of the EURAD project has been parallelized and tested on the Paragon XP/S. The model includes advanced modules for the numerical treatment of gas phase chemistry, transport and diffusion processes, radiation, cloud physics with aqueous phase chemistry, and aerosol processes.The model's integration domain encompasses Europe. Comprehensive chemistry-transport models are generally found to be well suited for massively parallel processing since the arithmetic-to-communication ratio is high. However this observation proofs insufficient to account for an efficient parallel performance. With increasing complexity of the model the local state of the atmosphere ensures very different branches of the modules' code where large differences in the runtime of processors occur. Emissions, changes in actinic flux, and all processes associated with cloud modelling are highly variable in time and space and therefore induce large load imbalances which severely affect the model efficiency. Based on a horizontal grid decomposition approach a method is proposed where the integration domain of the individual processors are locally adjusted to accomodate for load imbalances.This ensures a minimal communication volume with data exchange only to the next neighbours.

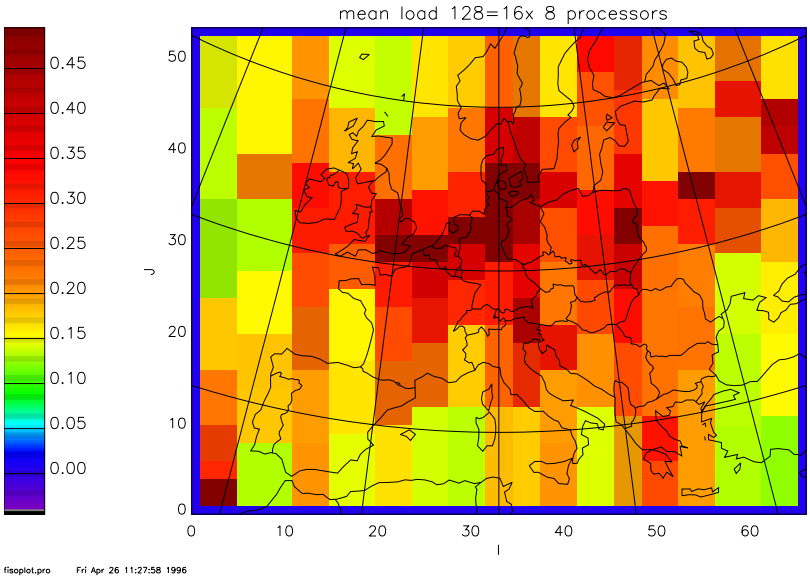


Figure 1: An application of $16 \times 8=128$ compute nodes, where the mean specific computational load of each processor is indicated by colour coding.

The load balancing strategy used here allows for an individual two-dimensional adjustment of the processors' subdomains.With this method more than four neighbour nodes are accepted by each inner node.The computational load of the processors are evaluated every timestep and the inner boundaries are then adjusted to evenly balance computation time for the next timestep.

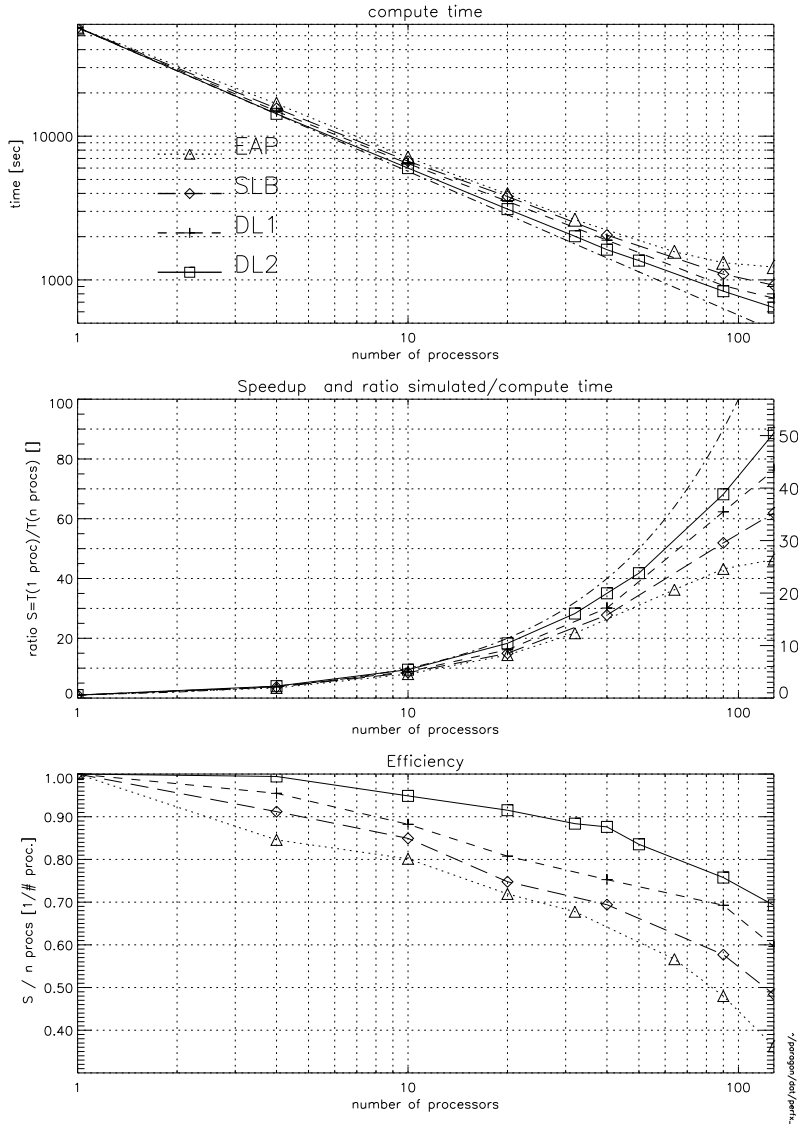


Figure 2: Performance results in terms of compute time, speedup, and efficiency, dependent on the number of processors and the load balancing strategy applied.

EAP: means trivial equal area partition; SLB: static load balancing with a four-neighbour-node restriction for the inner processors; DL1 dynamic load balancing, again with only four neighbour nodes; and DL2 with possibly more than four neighbour processors. The latter strategy is exposed in figure1. Due to dynamic load balancing an increase in efficiency from 36% to nearly 70% could be achieved.

References

1. Carmichael, D.M. Cohen, S-Y. Cho, and M.H. Ogutzun, Coupled Transport-Chemistry Calculations on the Massively-parallel Processor Computer, Comput. Chem. Eng. **13** (1989), 1065-1073
2. Dabdub, and S. Seinfeld, Air Quality Modelling on Massively-Parallel Computers, Atmos. Env. 28 (1994), 1679-1688

Symmetric Lanczos method for nonlinear structural Finite Element analysis

Ingrid Lenhardt, Thomas Rottner

University of Karlsruhe, Institute for Applied Mathematics/Institute for Mechanics

Project: VERSA (HPSC-Project, funded by BMBF)

Directors: G. Alefeld, K. Schweizerhof

Introduction. The nonlinear systems of equations arising in nonlinear structural finite element analysis are tackled usually using Newton-like methods. In every Newton step a large sparse system of linear equations has to be solved. For the latter in general direct methods are preferred over iterative solvers, because the condition number of the stiffness matrix is often large, which then reduces the convergence rate of iterative solvers. The bad conditioning is typical for some structural models, for example for thin shells, where the membrane stiffness is often considerably larger than the bending stiffness by a factor of magnitude. Also in computations of a whole load-deflection path the stiffness matrix of a structure can even become singular at so called limit points or bifurcation points and may remain indefinite on unstable solution paths.

Investigations of the authors on the application of the Lanczos algorithm [4] to such problems have shown that – with appropriate preconditioners – iterative methods can be developed which are robust and efficient. Though the real advantage of iterative solvers seems to exist on distributed memory machines, even on serial machines the performance can be improved compared with direct solvers while saving memory capacity. With a specific modification of the Lanczos algorithm in combination with arc-length procedures (see e.g. [5]) a further speed-up of the nonlinear analysis can be achieved.

The Lanczos algorithm. In [4] the following implementation of the Lanczos Algorithm for symmetric linear systems of equations is presented:

Phase I (Start): $x_0,\ r_1 = b - Ax_0,\ \vartheta_1 = M^{-1}r_1,$
$\beta_1 = \sqrt{r_1^T \vartheta_1},\ q_1 = \frac{r_1}{\beta_1},$
$q_0 = c_0 = d_0 = 0,\ \delta_0 = 0$

Iteration: for $j = 1, 2, \ldots$:

Phase II: Compute T_j
- (A) $u_j = \frac{\vartheta_j}{\beta_j}$
- (B) $z_j = Au_j$
- (C) $\alpha_j = u_j^T z_j$
- (D) $r_{j+1} = z_j - \alpha_j q_j - \beta_j q_{j-1}$
- (E) $\vartheta_{j+1} = M^{-1} r_{j+1}$
- (F) $\beta_{j+1} = \sqrt{r_{j+1}^T \vartheta_{j+1}}$
- (G) $q_{j+1} = \frac{r_{j+1}}{\beta_{j+1}}$

Phase III: LDL^T-factorisation of T_j

$$d_j = \alpha_j - \delta_j^2 d_{j-1}$$

$$\delta_{j+1} = \frac{\beta_{j+1}}{d_j}$$

Phase IV: Solve triangular systems,

$$\zeta_j = -\frac{\delta_j d_{j-1}}{d_j},\ (\zeta_1 = \frac{\beta_1}{d_1})$$

$$c_j = u_j - \delta_j c_{j-1}$$

$$x_j = x_{j-1} + \zeta_j g_j$$

Phase V: Stop if $\beta_{j+1}\zeta_j ||b|| < \varepsilon$

A parallel implementation. In phase II two dot products are computed, where global communication is necessary. The following reformulation of phase II allows overlapping of communication for the dot products with computation of matrix vector multiplication and preconditioning operations.

(A) $\psi_j = A\vartheta_j$

(B) $z_j = \frac{\psi_j}{\beta_j},\ u_j = \frac{\vartheta_j}{\beta_j}$

(C) $\alpha_j = u_j^T z_j$

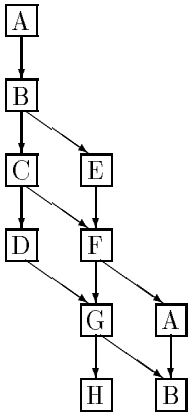
(D) $r_{j+1} = z_j - \alpha_j q_j - \beta_j q_{j-1}$

(E) $\varphi_j = M^{-1} z_j$

(F) $\vartheta_{j+1} = \varphi_j - \alpha_j u_j - \beta_j u_{j-1}$

(G) $\beta_{j+1} = \sqrt{r_{j+1}^T \vartheta_{j+1}}$

(H) $q_{j+1} = \frac{r_{j+1}}{\beta_{j+1}}$



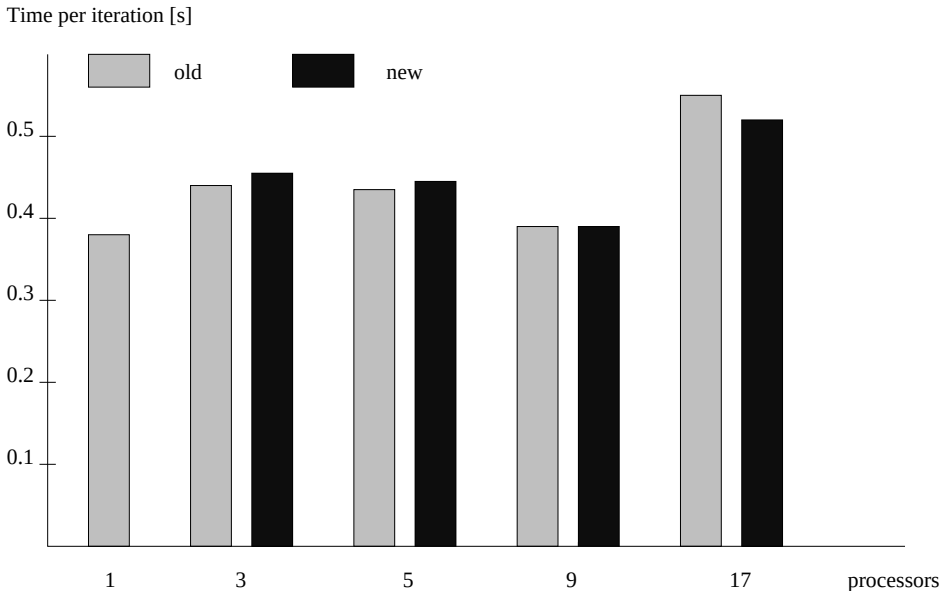


Figure 1: Time per iteration for old and new parallel implementation for an example from finite element structural analysis.

References:

1. R.Cook, A reformulation of preconditioned conjugate gradients suitable for a local memory multi-processor, in R.Beauwens, P.de Groen: Iterative Methods in Linear Algebra (Proceedings of the IMACS International Symposium on Iterative Methods in Linear Algebra, April 1991), 313-322
2. Lenhardt, I., Rottner, Th., Lösung nichtlinearer Gleichungssysteme aus der Strukturmechanik unter Einsatz einer parallelen Version des Lanczos Verfahrens}, Proc. Statustagung des BMBF "Stand und Perspektiven des Parallelen Höchstleistungsrechnens und seiner Anwendungen", Jülich (1995)
3. Lenhardt, I., Rottner, Th., Eine parallele Implementierung des Lanczos-Verfahrens für lineare Gleichungssysteme der Strukturmechanik, in preparation
4. Papadrakakis, M., Smerou, S., A new implementation of the Lanczos method in linear problems, Int. J. Numerical Meth. Eng. 29, 141-159 (1990)
5. Schweizerhof, K., Quasi-Newton Verfahren und Kurvenverfolgungsalgorithmen für die Lösung nicht-linearer Gleichungssysteme in der Strukturmechanik}, Universität Karlsruhe, Schriftenreihe des Instituts für Baustatik, 9 (1989)

4. Literature

[1] Berrendorf, R., Burg, H.C., Detert, U.
Leistungscharakteristika von Parallelrechnern: Fallstudie Intel Paragon
it+ti Informationstechnik und Technische Informatik 2/95, pp. 37-45

[2] Berrendorf, R., Burg, H.C., Detert, U., Esser, R., Gerndt, M., Knecht, R.
Intel Paragon XP/S – Architecture, Software Environment, and Performance
Internal Report KFA-ZAM-IB-9409

[3] Berrendorf, R., Gerndt, M., Mairandres, M.
Programming Shared Virtual Memory on the Intel Paragon Supercomputer
Internal Report KFA-ZAM-IB-9509

[4] Esser, R., Knecht, R.
Applications on KFA's Intel iPSC/860
Internal Report KFA-ZAM-IB-9218

[5] Hoßfeld, F.
Vector-Supercomputers
Parallel Computing 7 (1988), pp. 373

[6] Hoßfeld, F.
On the Supercomputational Background of the Research Centre Jülich
Internal Report KFA-ZAM-IB-9502

[7] Hoßfeld, F.
"Grand Challenges" – wie weit tragen die Antworten des Supercomputing?
Internal Report KFA-ZAM-IB-9117